ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO TÉCNICO DE TELECOMUNICACIÓN, ESPECIALIDAD EN SONIDO E IMAGEN

Título del proyecto:

"DESARROLLO DE HERRAMIENTA SOFTWARE PARA PROCESADO EN EL DOMINIO DEL TIEMPO DE SEÑALES PROVENIENTES DEL ANALIZADOR DE REDES"

Pablo Puig Usandizaga

Iñigo Ederra Urzainqui

Pamplona, 26 Junio 2013

Pablo Puig Usandizaga

Resumen

En este Proyecto Final de Carrera se ha implementado un software para el procesado en el dominio del tiempo de los datos obtenidos mediante un analizador vectorial de redes, incluyendo herramientas de postprocesado.

El software implementado puede ejecutarse en cualquier versión del software Matlab, ya que ha sido programado utilizando comandos universales, comunes a todas las versiones.

Este software implementa dos formas de procesado: el modo paso bajo y el modo paso banda. El proceso que sigue el software con el modo paso bajo, es el siguiente: obtiene los datos de los archivos provenientes del Analizador y los adapta al software. Estos datos se encuentran en el dominio frecuencial. Construye la señal a partir del módulo y de la fase. Una vez construida, calcula la parte negativa del espectro, teniendo en cuenta que debe verificar la simetría hermítica y concatena ambas señales en una única. A esta le aplica una ventana frecuencial de Kaiser-Bessel y *Zero Padding*. Para cambiar al dominio temporal implementa la Transformada Inversa de Fourier. En el dominio temporal ofrece la opción de poder aplicar distintas herramientas de postprocesado, como el *gating* o ventanas.
El modo paso banda sigue el mismo proceso con la única diferencia que construye la señal tan solo con el módulo y la fase, sin tener en cuenta la señal conjugada.

El software viene integrado en una interfaz gráfica, en la cual podremos observar la señal tanto en tiempo como en frecuencia, y en módulo y fase. También podremos variar los parámetros de *Zero Padding* y el tamaño de la ventana frecuencial de Kaiser-Bessel. Además de todo eso, podremos elegir los valores del *gating* temporal así como qué ventana temporal implementar.

Se han comprobado los resultados con los que proporciona la herramienta de procesado en el dominio del tiempo que proporciona el Analizador vectorial de redes.

Pablo Puig Usandizaga

Abstract

In my Research Project has been implemented a software for processing in the time domain the data captured using a VNA (vector network analyzer), including post-processing tools.

The implemented software can run on any version of Matlab software, because it has been programmed using universal commands, common to all versions.

This software implements two forms of processing: the low-pass mode and the bandpass mode. The process that the low-pass mode follows is: get the data from the file coming from the VNA and adapts them to the software. The data are in the frequency domain. It builds the signal from the module and from the phase. Once built, it calculates the negative part of the spectrum, considering that the signal must verify the Hermitian symmetry and concatenates the two signals into one. To this signal it applies a frequency window and Zero Padding. To change to the time domain implements the Inverse Fourier Transform. In the time domain it gives you the option to apply different postprocessing tools, such as gating or windows.
The bandpass mode follows the same process with the only difference that it constructed the signal with only the magnitude and the phase, without taking account of the conjugate signal.

The software is integrated into a graphical interface, in which we can observe the signal in time and in frequency, and in module and phase. We can also vary the parameters of Zero Padding and the size of the frequency window Kaiser-Bessel. Besides all that, we can choose the values of temporal gating and implement the window that we want.

The results have been checked with those that provided the processing tool int he time domain that provides the VNA.

Índice

1. Objetivos

A la hora de elegir el proyecto final de carrera tuve claro que quería algo útil y práctico, que sirviese para mejorar las operaciones o aplicaciones de forma más rápida o incluso en la medida de lo posible hacerlas más fáciles, que pudiese aportar algo nuevo y no fuese una réplica de algo ya existente, en definitiva deseaba innovar, crear. Al ofrecerme Iñigo, mi tutor, este proyecto, vi que solucionaba problemas o defectos que tiene el analizador de redes, y es por ello por lo que me decidí por este proyecto. Además de eso, el tema principal del proyecto es el análisis de señales, un tema que he trabajado a lo largo de mi carrera universitaria y siempre me ha gustado. Otro motivo que me impulsó en la elección del proyecto es que yo me considero una persona innovadora, curiosa y con ganas de desarrollar proyectos innovadores que me ayuden a crecer y formarme, tanto académica como personalmente, y este proyecto cumplía estas inquietudes.

El proyecto final de carrera trata de implementar un software para el procesado en el dominio del tiempo de los datos obtenidos mediante un analizador vectorial de redes, tanto en modo paso banda como en modo paso bajo, teniendo la opción de poder elegir en la interfaz gráfica el modo que deseemos. Lo que buscamos es independizarnos del Analizador, de forma que hagamos una única medida y luego procesemos en el dominio del tiempo los datos obtenidos.

Dicho software utilizará distintas herramientas de postprocesado como la implementación de diversos tipos de ventana, tanto temporales como frecuenciales, la visualización de su efecto en la respuesta en frecuencia, la obtención de parámetros de las señales, como el módulo y la fase, la selección del parámetro a procesar y la aplicación de un *gating temporal*. Todas estas herramientas vendrán integradas en una interfaz gráfica, de modo que el uso de dichas herramientas sea más sencillo y podamos jugar con todas las opciones que va a tener la interfaz.

He querido poner en práctica la frase que expresó Thomas Edison: “No he trabajado un solo día de mi vida, todo ha sido diversión”, tratando de disfrutar con este proyecto.

2. Introducción

En este apartado se describen los elementos más destacados del proyecto.

2.1 Parámetros S

Los parámetros de dispersión o parámetros S proporcionan una completa descripción de como se ve una red o dispositivo en sus N puertos.

Para la definición de una red multi-puerto, se asume que todos los puertos salvo el que se encuentra bajo consideración o el par de puertos bajo consideración, tienen una carga conectada a ellos idéntica a la impedancia del sistema y que cada puerto tiene asignado un entero 'n' que varía de 1 a N, donde N es el número total de puertos. Para un puerto n, la definición de parámetros-S asociados se realiza en función de 'ondas de potencia' incidente y reflejada, a_n y b_n respectivamente. Ondas de potencia son versiones normalizadas de las ondas viajeras de tensión incidente y reflejada correspondientes, V_n^+ y V_n^- respectivamente. Éstas están relacionadas con la impedancia del sistema Z_0 de la siguiente manera:

$$a_n = \frac{V_n^+}{\sqrt{Z_0}} \tag{2.1}$$

$$b_n = \frac{V_n^-}{\sqrt{Z_0}} \tag{2.2}$$

Para todos los puertos de la red, las ondas de potencia reflejadas pueden definirse en términos de la matriz de parámetros-S y las ondas de potencia incidentes a través de la siguiente ecuación:

$$\begin{pmatrix} b_1 \\ . \\ . \\ . \\ b_n \end{pmatrix} = \begin{pmatrix} S_{11} & S_{12} & . & . & S_{1n} \\ S_{21} & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ S_{n1} & . & . & . & S_{nn} \end{pmatrix} \begin{pmatrix} a_1 \\ . \\ . \\ . \\ a_n \end{pmatrix} \tag{2.3}$$

Los elementos de los parámetros-S se representan individualmente con la letra mayúscula 'S' seguida de dos subíndices enteros que indican la fila y la columna en ese orden de la posición del parámetro-S en la matriz de parámetros-S. [7]

Los parámetros S se miden con un Analizador vectorial de redes.

2.2 Analizador vectorial de redes

Es un instrumento de medición muy poderoso y flexible. Su capacidad básica es obtener los parámetros de transmisión y reflexión de señales, conocidos como parámetros-S. Concretamente, lleva a cabo medidas de respuestas de componentes de microondas y GHz en magnitud y fase en el dominio frecuencial. Esto proporciona datos valiosos para el desarrollo de un diseño o para comprobar el rendimiento de un dispositivo o sistema. Además, dicho instrumento incluye herramientas en el dominio del tiempo para mejorar la capacidad de análisis de los datos en el dominio del tiempo, como el *gating temporal* o distintos tipos de ventanas. [2]
Además de este tipo de analizador, existen los analizadores de redes escalares, cuya principal diferencia es que el vectorial mide las propiedades de módulo y fase, mientras que el escalar únicamente mide las propiedades de fase.

La respuesta de los dispositivos habitualmente se da en frecuencia. La información temporal nos da información de donde suceden los fenómenos que forman la respuesta en frecuencia, por lo que muchas veces es interesante tener información tanto en tiempo como en frecuencia, puesto que hay veces que no se observan partes de la señal en frecuencia pero si en tiempo. Esto se suele dar, por ejemplo, con las reflexiones, de forma que se pueden eliminar aquellas que no formen parte de la respuesta del componente. Esto ocurre habitualmente en medidas de antenas para eliminar efectos de reflexiones.

Existen Analizadores de distintas marcas, pero para hacernos a la idea de cómo es un Analizador de redes vectorial, se incluye una foto de uno de ellos.

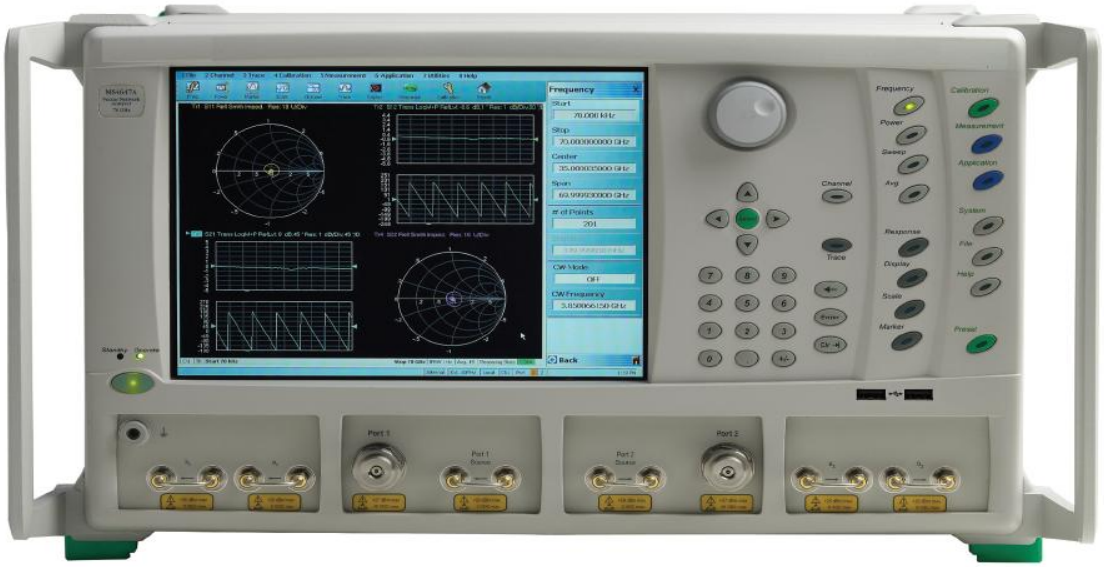

Figura 2.1: Analizador de redes vectorial.

2.2.1 Modos de procesado en dominio frecuencial

El Analizador implementa dos tipos de procesados en el dominio del tiempo, el modo paso bajo y el modo paso banda.

2.2.1.1 Modo paso bajo

El modo paso bajo en el dominio del tiempo simula una medición tradicional de TDR (Time Domain Reflectrometry). Hay algunas limitaciones específicas en el rango de frecuencias de la medición. Se requiere que los puntos de los datos positivos de las medidas estén linealmente espaciados de modo que estén relacionados desde la componente DC hasta la frecuencia de parada.
Las frecuencias medidas se deben establecer de modo que la frecuencia de parada sea igual al producto de la frecuencia de inicio y el número de punto. El analizador de redes vectorial tiene una función que realiza esto de forma automática. A partir de esto, el tiempo de subida está determinado por la pendiente máxima de la frecuencia más alta medida, aunque varía con el factor de la ventana.

También, puesto que la Transformada de Fourier incluye los efectos de la componente DC, y como el analizador de redes vectorial no mide la componente DC, el valor de la componente DC debe ser extrapolado. El resto de los datos se calcula a partir de tomar una imagen espejo de los datos originales medidos, en el que la respuesta en frecuencia negativa es el conjugado de la respuesta en frecuencia positiva y, por lo tanto, la respuesta en el dominio del tiempo es real pura (sin partes complejas).

Debido a que el modo paso bajo incluye la componente DC y la parte negativa de la señal, este modo tiene mejor resolución en el dominio temporal para un rango de frecuencias determinado que el modo paso banda.

2.2.1.2 Modo paso banda

El modo paso banda proporciona un método alternativo de transformación en el dominio del tiempo que se puede utilizar cuando en el modo paso bajo no puede cumplir la suposición de frecuencias relacionadas armónicamente. Esto ocurre, por ejemplo, en la medición de una red que está filtrada paso banda o paso alto.

La salida de una medición de un Analizador de redes vectorial es típicamente un número impar de puntos, espaciados linealmente. La Transformada Inversa de Fourier se calcula únicamente en los puntos medidos, siendo la respuesta en frecuencia negativa el conjugado de los datos medidos. El modo paso banda no supone una función frecuencial hermítica, sino que utiliza los datos como si la respuesta frecuencial fuera únicamente positiva.

El modo paso banda utiliza ventanas, donde el centro de la ventana es la frecuencia central del conjunto de datos. En cambio, el centro de la ventana en el modo paso bajo es la componente DC o el primer punto del conjunto de datos.

La Transformada Inversa en el modo paso banda se define como:

$$f_{BP}(t) = \frac{1}{W_0}\frac{\Delta\omega}{2\pi} \cdot \sum_{n=-N/2}^{N/2} F_{BP}(\omega_C + n\Delta\omega) \cdot W(n\Delta\omega) \cdot e^{j(\omega_C + n\Delta\omega)\,t} \qquad (2.4)$$

La respuesta del Analizador de redes en el modo paso banda devuelve siempre una respuesta compleja en el dominio del tiempo. Este efecto es debido a la eliminación de la suposición que la respuesta en frecuencia contiene elementos frecuenciales negativos.

Una de las consecuencias del modo paso banda es que la resolución es la mitad que en el modo paso bajo. [2]

El modo paso banda viene integrado en el analizador de redes vectorial como predeterminado, teniendo la opción de cambiarlo a modo paso bajo.

2.2.2 Gating temporal

Gating temporal se refiere al proceso de selección de una región de interés en una porción del dominio temporal, eliminando las respuestas no deseadas. Es por esto por lo que es una herramienta muy utilizada para el postprocesado de señales en el dominio del tiempo.

También se puede definir como la multiplicación del dominio temporal por una función matemática con un valor de uno sobre la región de interés y un valor de cero en el resto. Una vez aplicado el *gating temporal* se puede transformar al dominio frecuencial, sin el efecto de las otras respuestas temporales. Sin embargo, los efectos del "gating" son algo sutiles en su respuesta y hay consecuencias evidentes en la señal.

El *gating temporal* es un proceso fuerte, ya que su respuesta en frecuencia no está limitada por los datos medidos. Por lo que su respuesta temporal tiene transiciones bruscas y por consiguiente su respuesta frecuencial también tendrá transiciones bruscas. Es por esto que el *gating temporal* requiere de un enventanamiento de la señal antes de ser transformado al dominio frecuencial.

El "gating" es esencialmente un filtro en el dominio del tiempo. [2] [3] [4]

En la Figura que se muestra a continuación, se observa una señal en el dominio del tiempo a la que se le ha aplicado un *gating temporal* en un intervalo de tiempo de 0 a 0,2.

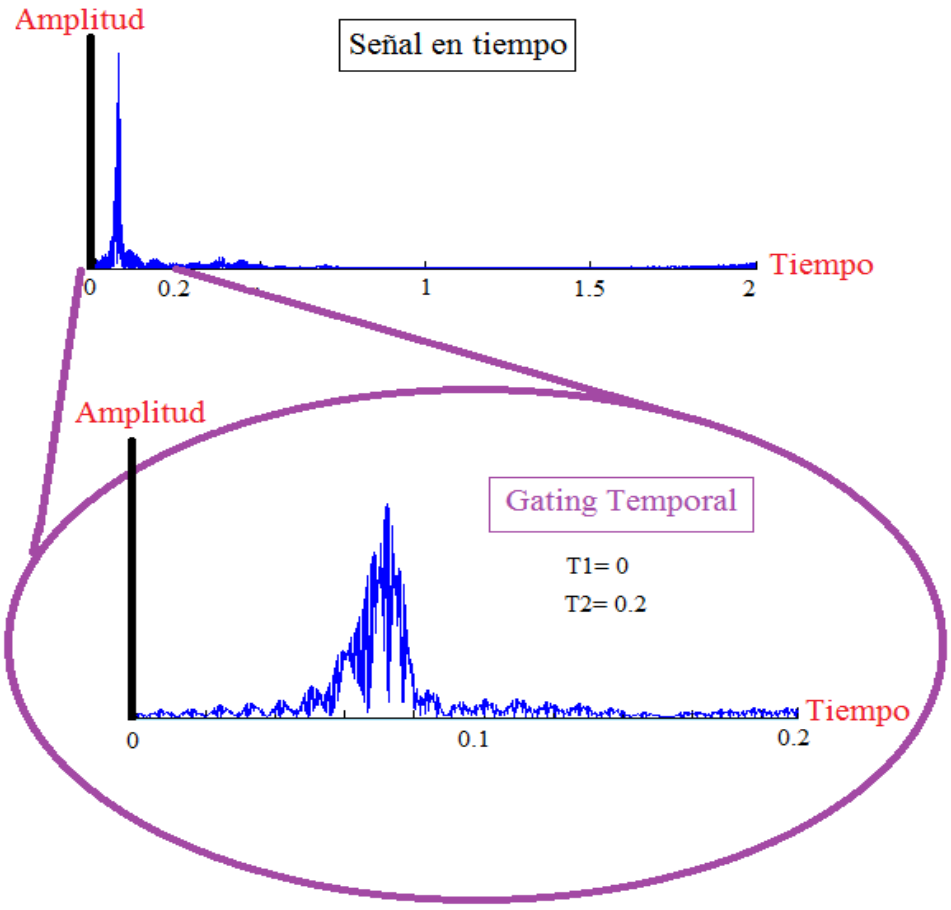

Figura 2.2: Ejemplo de aplicar un gating temporal a una señal en un intervalo de tiempo de [0,0.2].

2.2.3 Enventanados

Enventanar una señal significa multiplicar esta señal por otra (ventana), con el fin de suavizar esta, eliminando discontinuidades. Se puede enventanar una señal tanto en tiempo como en frecuencia.

2.2.3.1 Ventanas frecuenciales

Idealmente, las mediciones en el dominio frecuencial serían continuas sobre un rango de frecuencias infinita. Debido a que el Analizador de redes vectorial es capaz de medir en un rango de frecuencias finitas, se han desarrollado métodos para ayudar a lidiar con las limitaciones del mundo real. Una característica que está diseñada para mejorar las mediciones en el dominio del tiempo del Analizador de redes vectorial es el enventanamiento. El enventanamiento mejora el rango dinámico de la medición del dominio temporal mediante la modificación (filtrado) de los datos en el dominio frecuencial antes de la conversión al dominio temporal para producir un impulso con lóbulos laterales más bajos. Esto mejora en gran medida la eficacia en la visualización de las respuestas del dominio temporal que son muy diferentes en magnitud. Sin embargo, la reducción del lóbulo lateral se consigue con la compensación del aumento de la anchura del impulso.

Debido al limitado ancho de banda del sistema de medición, hay transiciones bruscas en las mediciones en el dominio frecuencial en las frecuencias de inicio y de fin. Es este rango limitado el causante de picos y rizado en la respuesta al impulso en el dominio temporal sin enventanar. Esta respuesta al impulso tiene dos efectos que limitan la utilidad de la respuesta en el dominio temporal.

En primer lugar, la anchura del impulso finito, causado por la naturaleza de la banda limitada del sistema de prueba, limita la capacidad de resolver entre dos respuestas muy próximas entre sí. La anchura del impulso es inversamente proporcional al rango de frecuencias de las mediciones, y la única manera de que la anchura del impulso pueda hacerse más estrecha es aumentar el rango de frecuencias.

En segundo lugar, los lóbulos laterales del impulso, causados por el punto de corte brusco en la frecuencia de parada, limitan el rango dinámico de la medición en el dominio temporal al ocultar las respuestas de bajo nivel dentro de los lóbulos laterales de las respuestas adyacentes de nivel superior.

Si los lóbulos laterales resultantes son demasiado altos, se pueden ocultar las respuestas más pequeñas del dispositivo bajo prueba y limitar el rango dinámico de la medición del dominio temporal. Una ventana se puede aplicar para modificar los datos en el dominio frecuencial, controlando de este modo los lóbulos laterales creados durante el proceso de truncamiento. Esto hace que la respuesta sea más útil en el aislamiento y en la identificación de las respuestas individuales.

Si bien el proceso de enventanar tiende a reducir la nitidez de la respuesta original reduciendo así el rizado en el dominio del tiempo, pudiendo causar un aumento en la anchura del impulso. Como se ha mencionado anteriormente, la anchura del impulso finito (o tiempo de subida) limita la capacidad de resolver entre dos respuestas muy próximas entre sí y los efectos de la anchura del impulso finito no puede mejorarse sin aumentar el intervalo de frecuencia de las mediciones. [7]

El analizador de redes vectorial implementa la ventana frecuencial de Kaiser-Bessel, cuya fórmula matemática se muestra a continuación:

$$w_k = \frac{I_0(\pi\alpha\sqrt{1-(2k/n-1)^2})}{I_0(\pi\alpha)} \qquad (2.5)$$

Donde 'I_0' es la función de Bessel de orden 0, cuyo valor está relacionado con el valor de 'α', que es un número real arbitrario que determina la forma de la ventana. La variable 'n' es un número natural que determina el tamaño de la ventana.
En función del valor de 'α' la ventana obtiene una forma u otra. En la siguiente Figura observamos cómo cambia la forma en función de dicha variable.

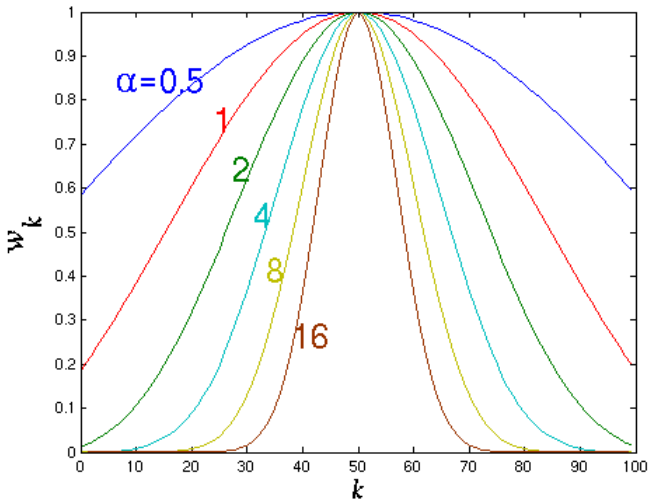

Figura 2.3: Diferentes formas de la ventana de Kaiser-Bessel según la variable 'α'.

2.2.3.2 Ventanas temporales

Una ventana temporal se implementa en el dominio temporal, con el fin de suavizar la señal del espectro. A la hora de escoger una ventana temporal, se requiere conocer sus características primero.

En la siguiente tabla se muestran las características de las ventanas temporales implementadas en el proyecto.

Ventanas	Ancho en -3 dB del lóbulo principal	Ancho en -6 dB del lóbulo principal	Máximo nivel del lóbulo lateral (dB)	Tasa de disminución del lóbulo lateral (dB/década)
Hamming	1.30	1.81	-43	20
Triangular	1.28	1.74	-27	40
Hanning	1.44	2.00	-32	60
Blackman	1.64	2.30	-58	60
Rectangular	0.88	1.21	-13	20

Tabla 2.1: Características de las ventanas utilizadas.

Cada ventana posee características que la hacen más o menos apropiada dependiendo de la aplicación. El primer paso para escoger apropiadamente una ventana es conocer o al menos tener una idea de cómo es el contenido espectral de la señal a la cual se le aplicará la ventana.

En las señales reales la interferencia es parte de las señales. Cuando hay intensas componentes de interferencia lejos de la frecuencia de interés, se escoge una ventana con alta tasa de disminución de lóbulo lateral. En caso de que la interferencia esté cercana a la frecuencia de interés, se escoge una ventana con bajo nivel máximo del lóbulo lateral. Cuando la frecuencia de interés este acompañada de varias componentes muy cercanas entre sí, la resolución espectral cobra relevancia. Se debe escoger una ventana con un muy angosto lóbulo principal. Cuando la precisión en la medida de amplitud de una componente es de mayor importancia que su ubicación, se elige una ventana con un lóbulo principal ancho.

Si el espectro de la señal es allanado o de banda ancha, se utiliza la ventana rectangular. La ventana de Hanning posee muy buena resolución en frecuencia y pocas pérdidas de información en el espectro. [8]

La preferencia entre una ventana y otra para determinada aplicación no tiene una regla general, se recomienda aplicar diferentes tipos de ventanas y experimentar hasta encontrar la mejor para cada aplicación particular.

En la siguiente tabla se muestran las fórmulas matemáticas de las ventanas temporales implementadas.

Nombre de la Ventana	**Secuencia en el dominio temporal h(n), 0 < n < M-1**
Hamming	$0.54 - 0.46\,Cos\,\frac{2\pi n}{M-1}$
Triangular	$1-\frac{2\lvert n\rvert}{M-1}$
Hanning	$\frac{1}{2}\left(1-Cos\,\frac{2\pi n}{M-1}\right)$
Blackman	$0.42 - 0.50\,Cos\,\frac{2\pi n}{M-1} + 0.08\,Cos\,\frac{4\pi n}{M-1}$
Rectangular	1

Tabla 2.2: Fórmulas matemáticas de las ventanas utilizadas.

A continuación, se detallan las ventanas temporales que se van a implementar.

2.2.3.2.1 Ventana de Hamming

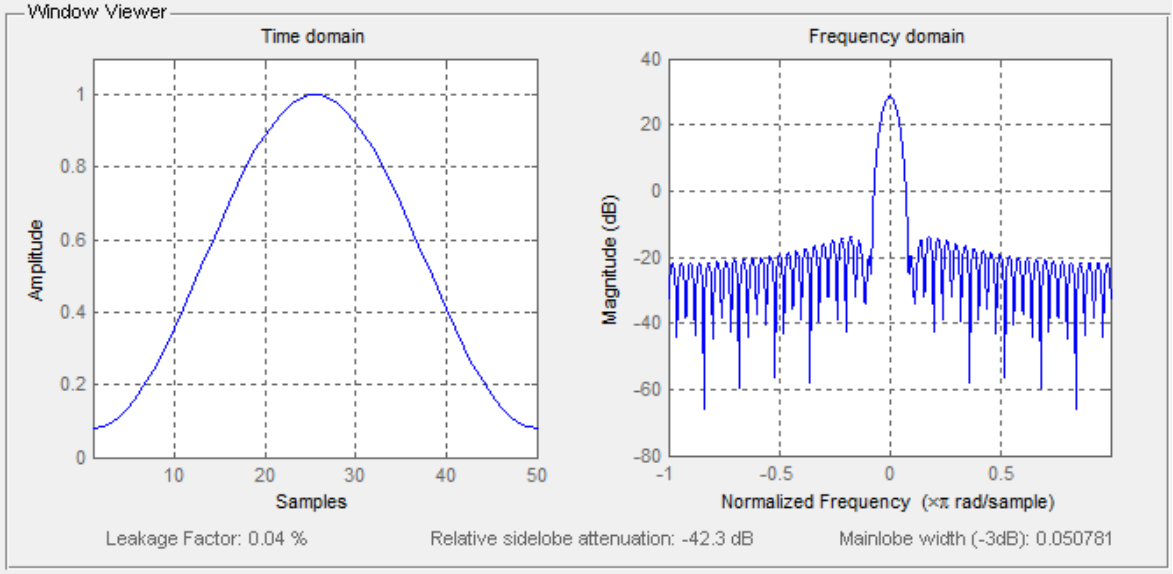

Figura 2.4: Ventana de Hamming en el dominio del tiempo y en el dominio de la frecuencia.

La ventana de Hamming posee un lóbulo principal muy ancho en el espectro. Realiza un gran trabajo reduciendo las pérdidas de información en el espectro.

2.2.3.2.2 Ventana Triangular

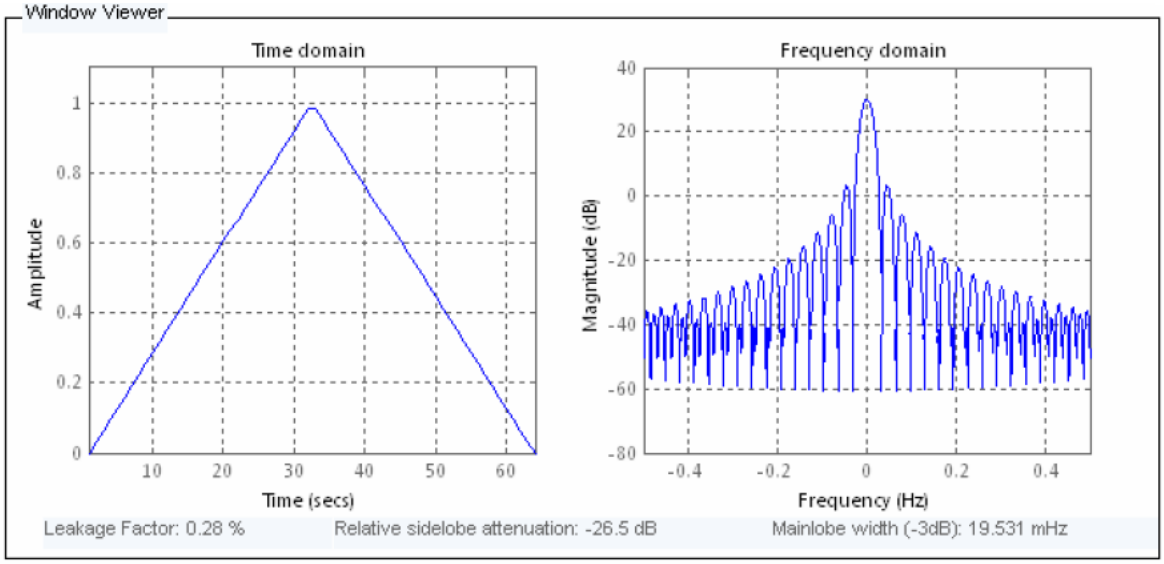

Figura 2.5: Ventana Triangular en el dominio del tiempo y en el dominio de la frecuencia.

La ventana Triangular posee un lóbulo principal en el espectro, similar al de la ventana de Hamming, pero con lóbulos laterales grandes. Hace un buen trabajo reduciendo pérdidas de información en el espectro. Esta ventana es una convolución de dos ventanas rectangulares.

2.2.3.2.3 Ventana de Hanning

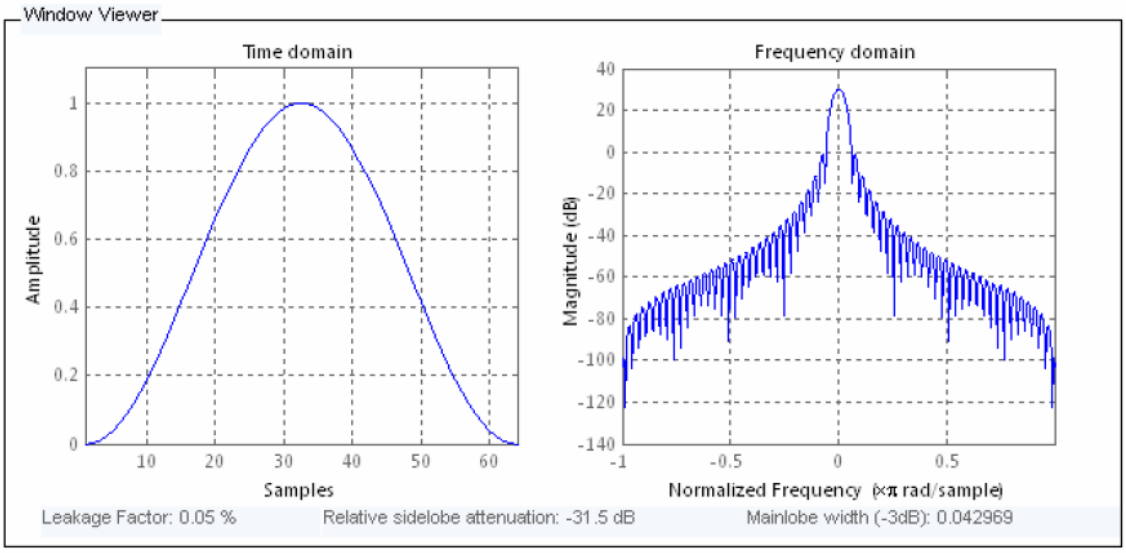

Figura 2.6: Ventana de Hanning en el dominio del tiempo y en el dominio de la frecuencia.

La ventana de Hanning posee un buen pico de lóbulo principal que permite tener un espectro de buena resolución, brinda además, muy buena reducción de pérdidas de información no deseadas en el espectro.

Esta ventana se basa en una combinación de dos ventanas más simples; la suma de una rectangular y una coseno. Mejora el decaimiento de altas frecuencias, aunque posee un nivel máximo de lóbulo lateral grande.

2.2.3.2.4 Ventana de Blackman

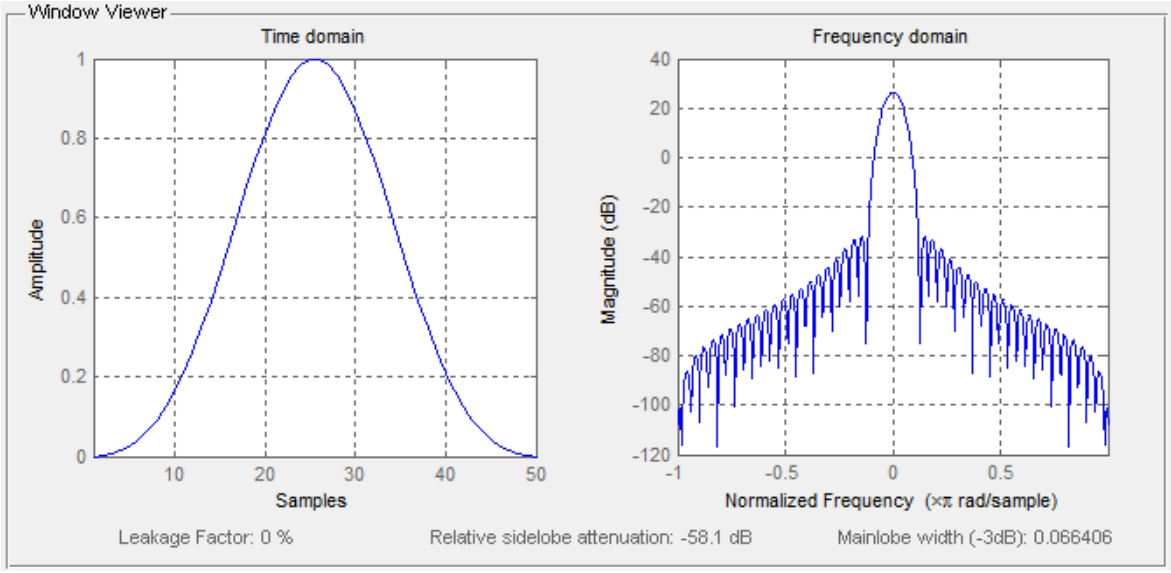

Figura 2.7: Ventana de Blackman en el dominio del tiempo y en el dominio de la frecuencia.

Es una ventana que posee un gran lóbulo principal en el espectro y lóbulos laterales bastante pequeños. El efecto de esta se asemeja bastante a la ventana de Hanning.

2.2.3.2.5 Ventana Rectangular

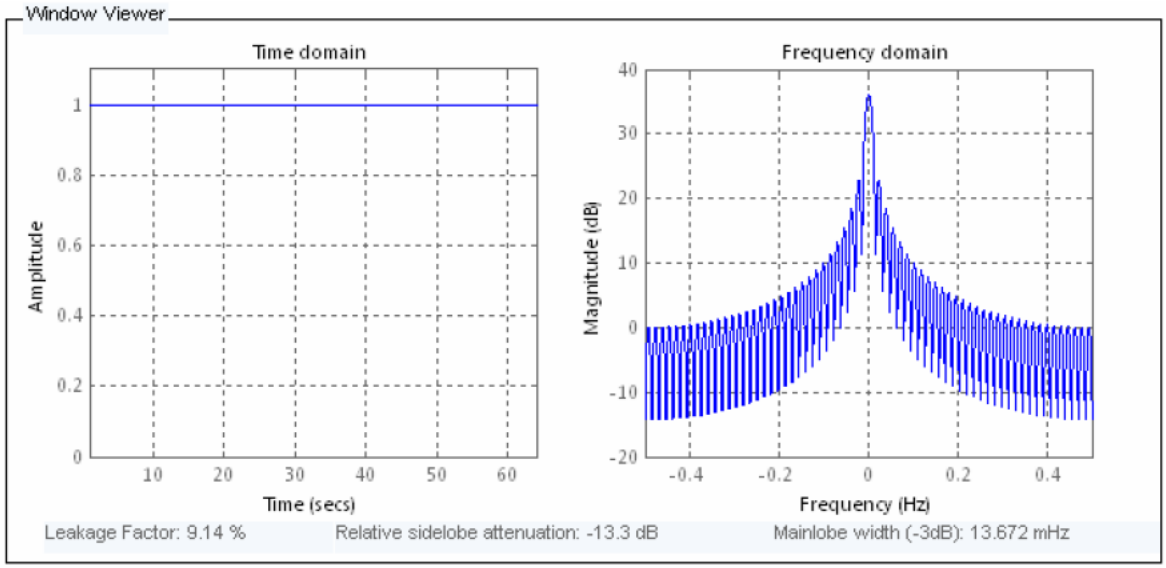

Figura 2.8: Ventana Rectangular en el dominio del tiempo y en el dominio de la frecuencia.

Es la ventana que se utiliza por defecto cuando se trunca una señal para obtener su espectro.

2.3 Transformada Discreta de Fourier

Dado que las señales que se van a procesar son discretas y periódicas y con un número de muestras finito, se va a implementar la Transformada Discreta de Fourier (DFT) para cambiar del dominio temporal al frecuencial y la Transformada Inversa (DFTI) para cambiar del dominio frecuencial al temporal.

2.3.1 Definición

Sea una señal discreta periódica x_k. Si el periodo fundamental lo denotamos por N, podemos escribir x (k+nN) = x_k para n = 0,1…. Como toda la información de la señal se encuentra en el intervalo k = 0,…, N-1 (N términos) lo que nos planteamos es la transformación de este vector temporal en otro espacio o dominio frecuencial mediante la expansión de series de Fourier, definido por la siguiente expresión de síntesis:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N}kn} \qquad k = 0, \ldots, N-1 \tag{2.6}$$

, donde x_n son los coeficientes de la amplitud y la exponencial las fases correspondientes en el dominio de la frecuencia.

Es importante remarcar que al utilizar la DFT se supone, de forma implícita, que la señal temporal es periódica con periodo N. Esta periodicidad no se debe ignorar, ya que muchos de los problemas de la DFT se deben a dicha repetición de la señal X_k, como por ejemplo el derrame espectral.

La Transformada Discreta de Fourier Inversa o DFTI queda definida como:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{\frac{2\pi i}{N}kn} \qquad n = 0, \ldots, N-1 \tag{2.7}$$

Destacar que los coeficientes del desarrollo forman una secuencia discreta de periodo N. [7]

2.3.2 *Zero Padding*

El *Zero Padding* o rellenado de ceros consiste en la adición de ceros a la secuencia temporal ya sea al comienzo de esta o al final o en ambas. Con esto, conseguimos aumentar la resolución frecuencial sin modificar el espectro de la señal. Esto es debido a que la Transformada Discreta de Fourier se calcula sobre más muestras, pero con el mismo margen frecuencial.

A continuación, vemos un ejemplo de una secuencia temporal de longitud N a la que añadimos al final de esta M ceros.

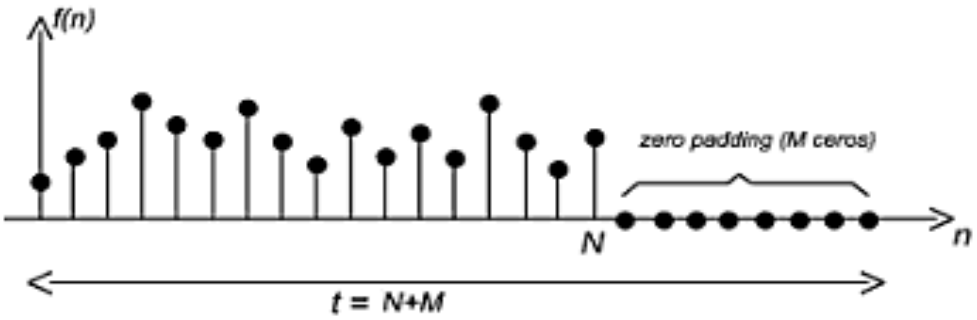

Figura 2.9: Adición de ceros al final de una secuencia temporal.

El Zero Padding, también se puede aplicar a una secuencia frecuencial, consiguiendo aumentar la resolución temporal. Esta adición de ceros se realiza en el medio, como observamos en la Figura siguiente:

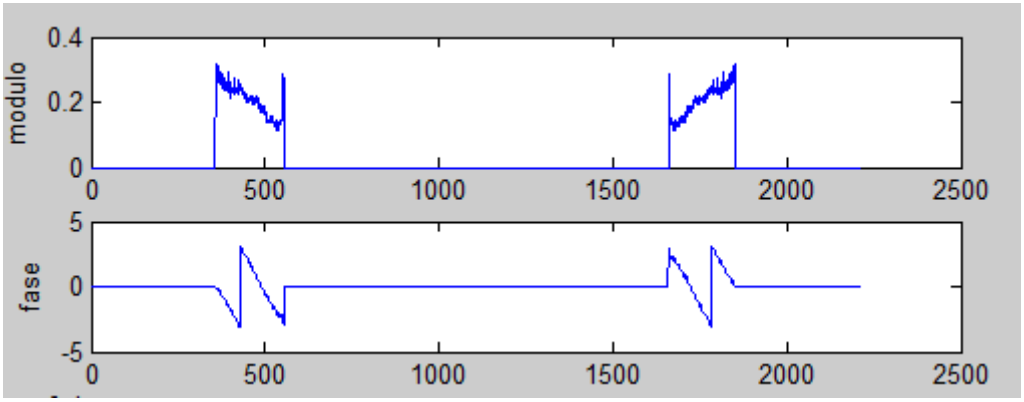

Figura 2.10: Adición de ceros en el medio de una secuencia frecuencial, tanto en el módulo como en la fase.

3. Desarrollo del software

Para el desarrollo del trabajo, lo primero es indicar los equipos necesarios para la realización del proyecto final de carrera, los cuales son: la herramienta software Matlab y el Analizador vectorial de redes.

El proyecto se divide en dos partes, cada una de las cuales está dedicada a un modo de trabajo del Analizador vectorial de redes . El modo paso bajo y el modo paso banda. De tal manera que como ocurre en un analizador de redes vectorial sea posible procesar la señal en ambos modos. Cada modo tiene una estructura distinta.

3.1 Modo paso bajo

El modo paso bajo está estructurado en cuatro grandes bloques, ordenados según el orden que sigue el software.

A continuación, en la Figura 3.1 he querido incluir un diagrama de flujo del modo paso bajo basándome según el orden que sigue el software, para que quede clara la estructura del proyecto así como la de la interfaz.

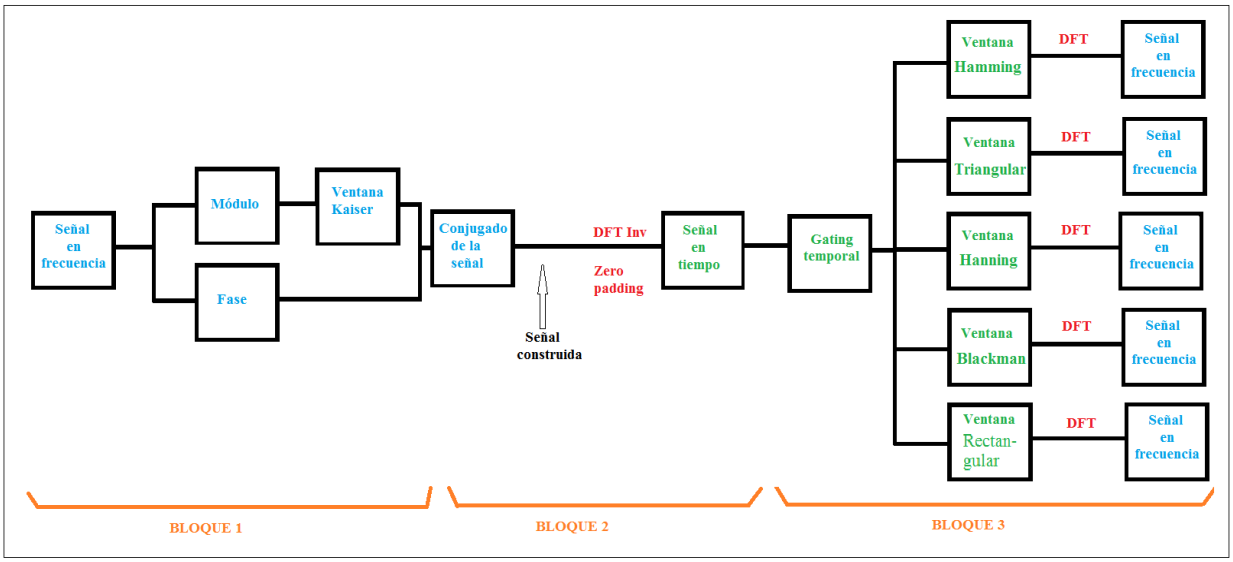

Figura 3.1: Diagrama de flujo del modo paso bajo. (El color azul indica el dominio frecuencial y el color verde indica el dominio temporal).

El primer bloque, consiste en adaptar el resultado del primer bloque a las características de los ficheros de datos generados por el Analizador de redes vectorial. Es de particular importancia automatizar la generación correcta de las escalas frecuenciales.
Este bloque engloba la lectura de los ficheros del analizador en formato .csv, la extracción de datos de dichos ficheros para obtener la información necesaria y la adaptación de dichos datos a la herramienta software.

Para la extracción de datos de dichos ficheros para obtener la información necesaria he tenido que crear una función para calcular el número de datos totales. Con esta función creada y sabiendo que la primera fila son las frecuencias, la segunda son los módulos y la tercera son las fases, podemos extraer los tres parámetros de entrada, a partir de las cuales construiremos las señales. En la adaptación de dichos datos a la herramienta software, el módulo hay que pasarlo de decibelios a lineal y la fase de grados a radianes.

El segundo bloque consiste en implementar la Transformada Discreta de Fourier Inversa para señales paso bajo. Esta implementación se ha llevado a cabo de la forma más genérica posible.
Para ello, lo primero que hago es construir la señal que va a ir desde la primera frecuencia del archivo hasta la última.

Se obtiene la siguiente señal:

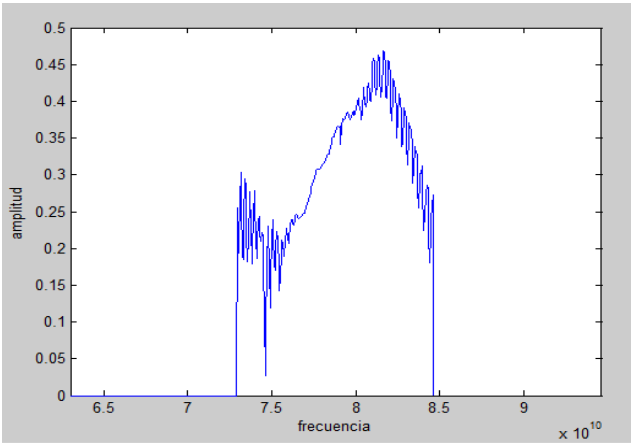

Figura 3.2: El módulo de la señal construida con los datos obtenidos del archivo.

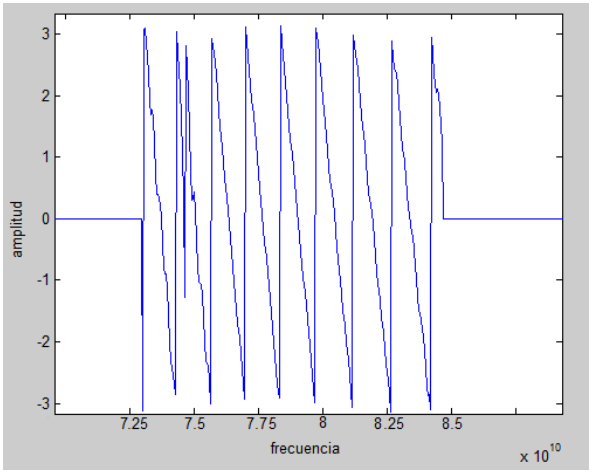

Figura 3.3: La fase de la señal construida con los datos obtenidos del archivo.

El segundo paso, es calcular la parte negativa del espectro, teniendo en cuenta que debe verificar la simetría hermítica, y después concatenar ambas señales en un único vector.

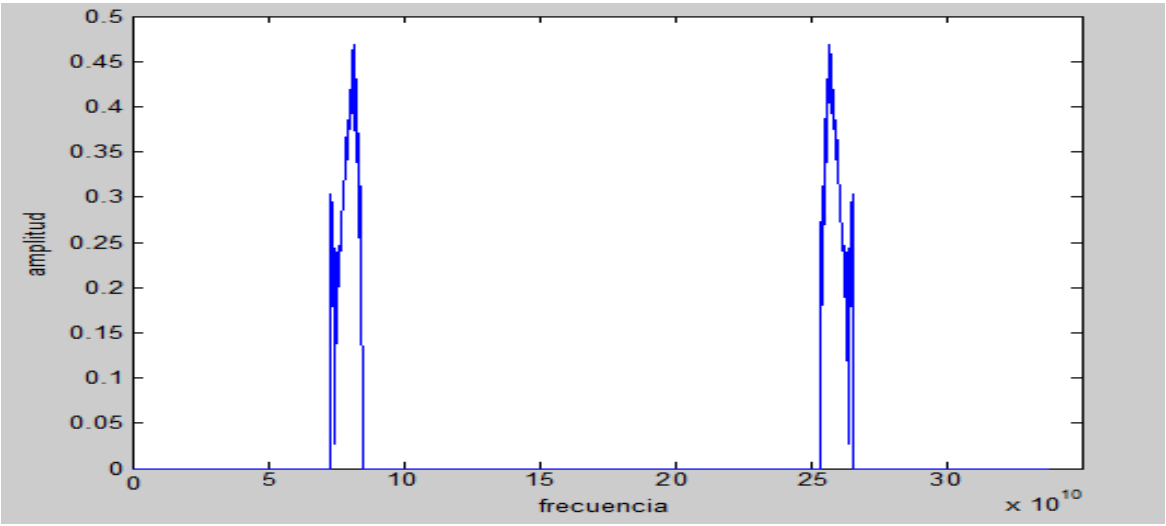

Figura 3.4: El módulo de la señal construida junto con su conjugada.

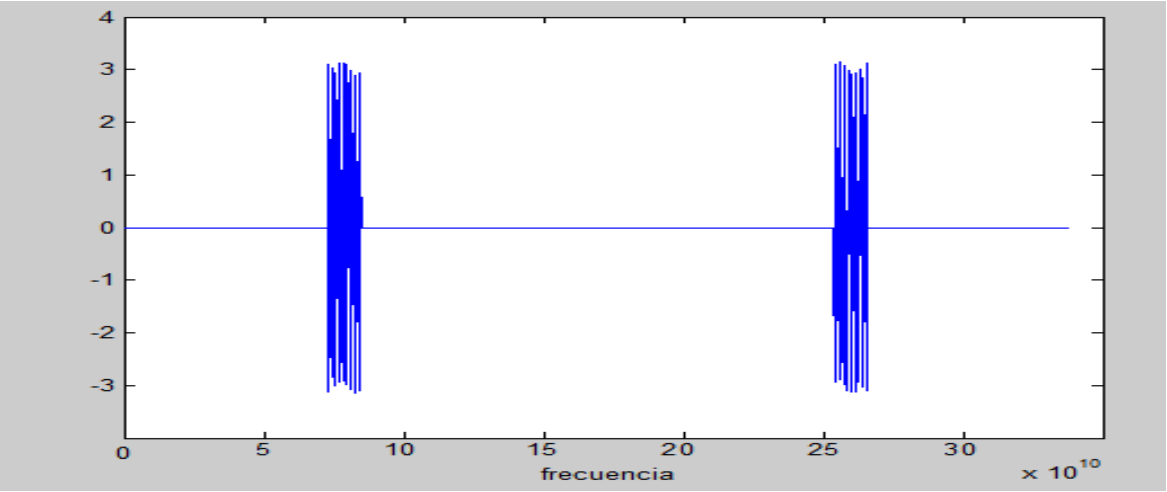

Figura 3.5: La fase de la señal construida junto con su conjugada.

En las Figuras anteriores se ha podido observar la señal construida, tanto en módulo como en fase . La parte negativa de la señal es el segundo pico de las Figuras, cumpliendo la simetría hermítica.

Por último, calculo la Transformada Discreta de Fourier Inversa de la señal.

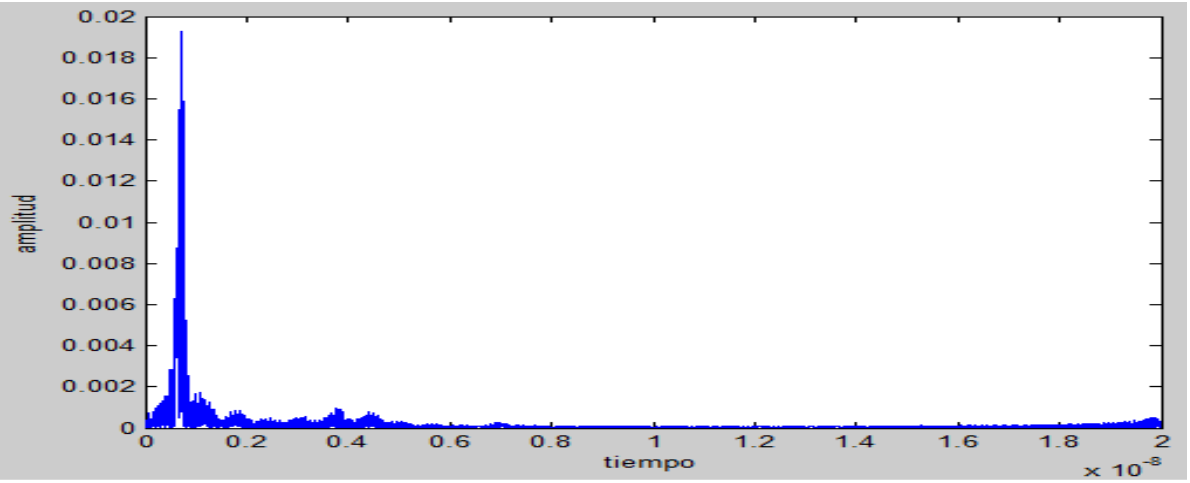

Figura 3.6 La Transformada Discreta de Fourier Inversa de la señal construida anteriormente.

El tercer bloque consiste en implementar las distintas opciones de procesado en el dominio del tiempo y en el dominio frecuencial. En particular, se consideran aquellas ya disponibles en el propio analizador de redes: uso de distintos tipos de ventanas, aplicación de *gating temporal* y la obtención de la respuesta en frecuencia de este caso.

Empezando con la implementación de la ventana de Kaiser-Bessel en el dominio frecuencial sobre la señal construida con los datos del archivo del analizador, para la cual introduciremos desde la interfaz el valor de α a implementar.

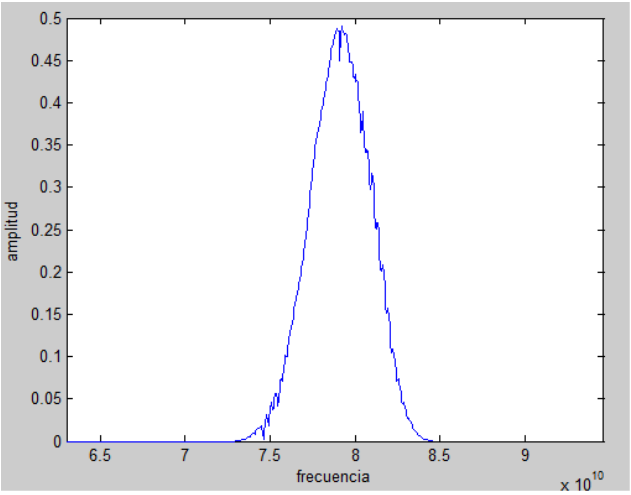

Figura 3.7: Señal con la ventana frecuencial aplicada.

Siguiendo con el *gating temporal*, para el cual introducimos dos parámetros de entrada desde la interfaz que serán los parámetros que definan el *gating temporal*, ya que toda la señal que se encuentre fuera de estos valores valdrá cero.

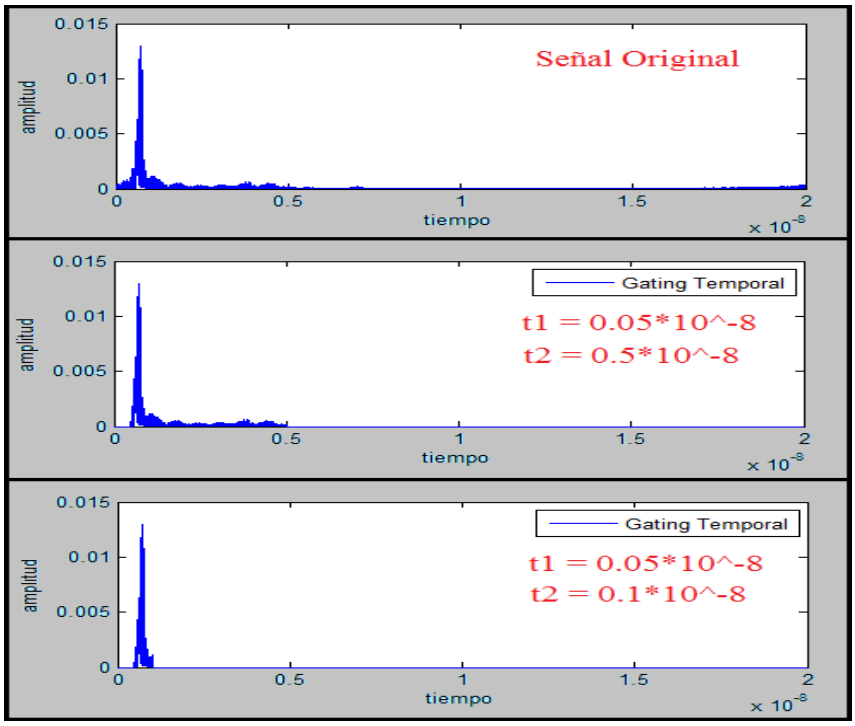

Figura 3.8: Ejemplo de una señal con distintos "gating temporales" aplicados.

A continuación, aplico los distintos tipos de ventana temporal sobre la señal con el *gating temporal* aplicado. Las ventanas utilizadas son: Hamming, Triangular, Hanning, Blackman y Rectangular. El proceso a seguir para aplicar las distintas ventanas temporales sobre la señal es el mismo para todas ellas. Se trata de multiplicar la señal por la ventana.
Por último, se calcula la respuesta en frecuencia de cada ventana, es decir, el espectro de cada señal enventanada.

El cuarto y último bloque, trata del desarrollo de una interfaz gráfica que permite el uso sencillo de las herramientas, incluyendo gráficas para la visualización de los resultados y botones diversos para la elección de las herramientas que se requiera, como por ejemplo, un tipo de ventana o un *gating temporal* concreto, así como la elección del fichero.

El software incluye muchas herramientas de implementación, como comandos, que simplifican y facilitan el código a programar. Todos los comandos que he utilizado vienen descritos en el Apéndice 2.

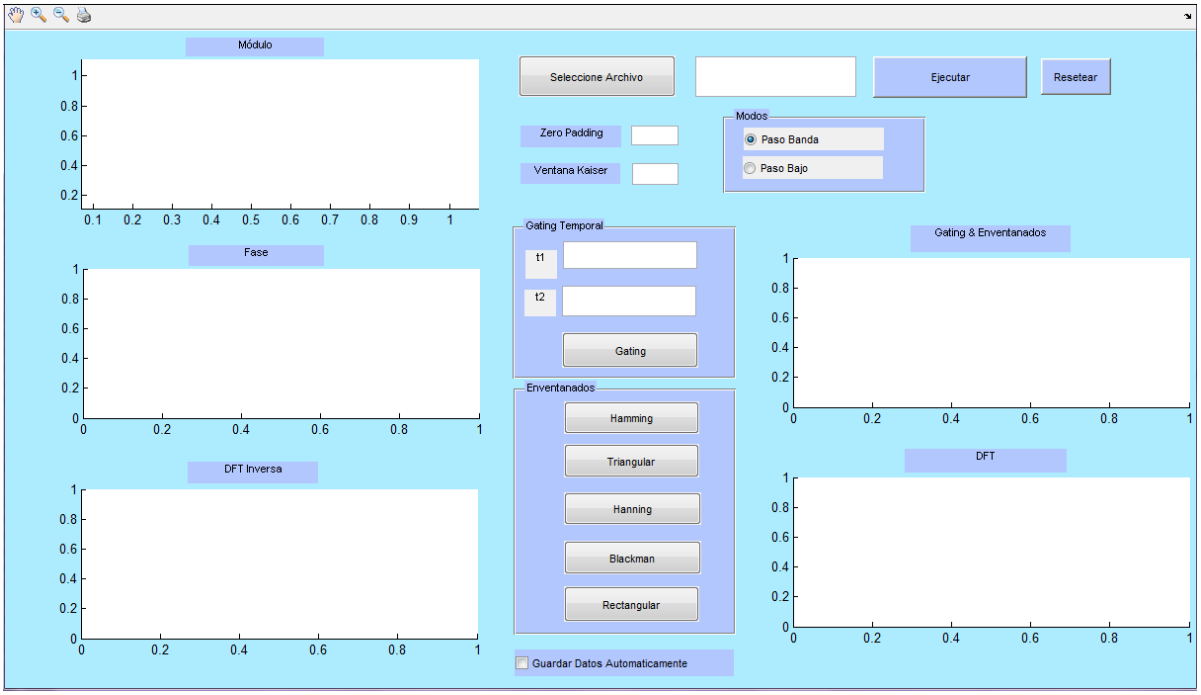

Figura 3.9: Interfaz gráfica del proyecto.

Observamos en la Figura 3.9, que es una interfaz gráfica muy intuitiva en la que todo resultado queda mostrado en una gráfica, con la posibilidad de guardar los datos y la misma gráfica en un archivo adjunto, accionando el botón de guardar los datos automáticamente.
Tenemos la posibilidad de analizar todos los archivos que queramos, seleccionando cada uno de ellos. Además, desde la interfaz se da la opción de introducir el *Zero Padding* y el valor de α de la ventana frecuencial Kaiser-Bassel, lo cual es muy útil, ya que podemos introducir el valor que más se adecue a nuestro interés. Pasa lo mismo con el *gating temporal*, los valores del rango temporal se introducen desde la interfaz, pudiendo cambiar estos valores tantas veces como se desee.

Figura 3.10: Ejemplos de valores asignados al Zero Padding, a la ventana frecuencial de Kaiser-Bessel y al gating temporal.

Si se desea aplicar una ventana temporal sobre una señal en el dominio del tiempo no hay más que pulsar sobre el botón de la ventana que se desee. Incluso, podemos ir pulsándolas una a una y ver cuál es la más apropiada para la señal.
A la hora de utilizar el software desde la interfaz se siguen los pasos descritos en el apartado Manual de Usuario.

3.2 Modo paso banda

El modo paso banda está estructurado de manera similar al modo paso bajo, descrito anteriormente. A continuación, en la figura 3.11 he querido incluir un diagrama de flujo del modo paso banda basándome según el orden que sigue el software, para que quede clara la estructura del proyecto así como la de la interfaz.

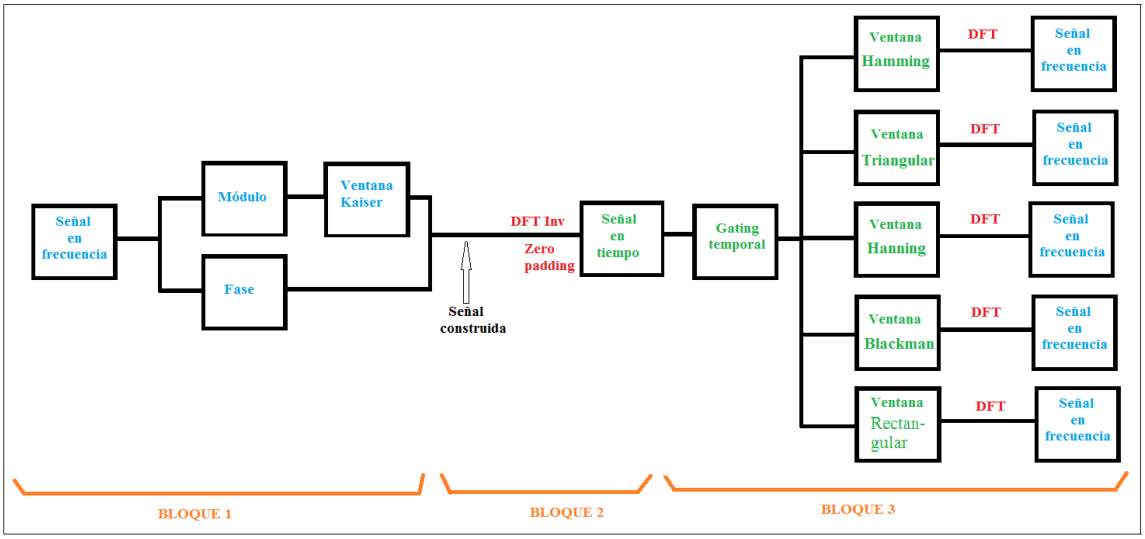

Figura 3.11: Diagrama de flujo del modo paso banda. (El color azul indica el dominio frecuencial y el color verde indica el dominio temporal).

El primer, tercer y cuarto bloque son exactamente igual que en el modo paso bajo, mismos pasos, mismas opciones y mismos métodos.

El segundo bloque, que es el que cambia respecto al modo paso bajo, consiste en implementar la Transformada Discreta de Fourier Inversa para señales paso banda.
Lo primero que hago es construir la señal.

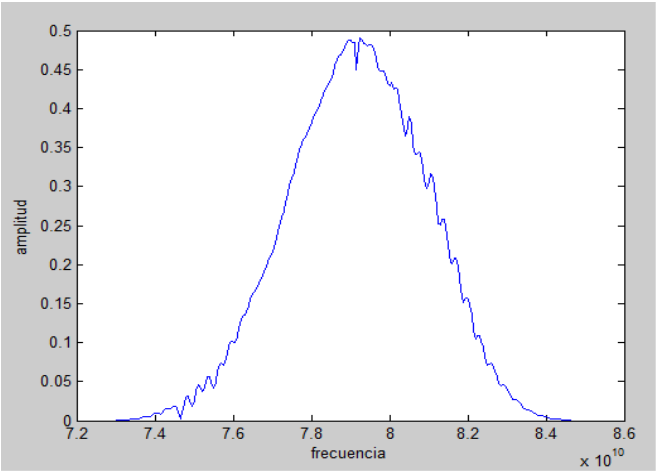

Figura 3.12: Módulo de la señal en modo paso banda.

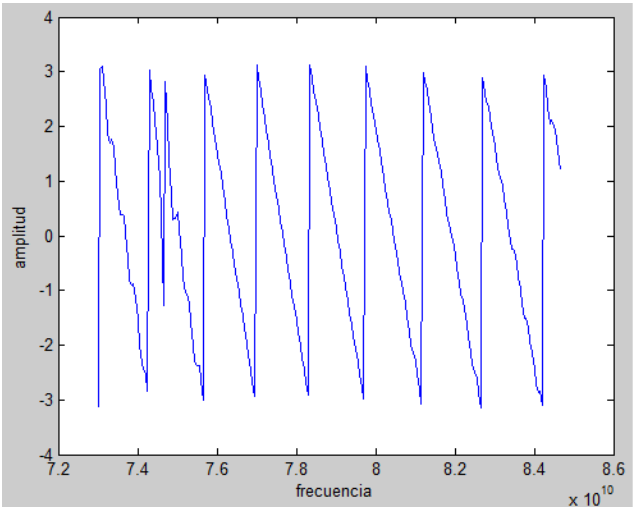

Figura 3.13: Fase de la señal en modo paso banda.

Lo siguiente, es calcular la Transformada Discreta de Fourier Inversa de la señal.

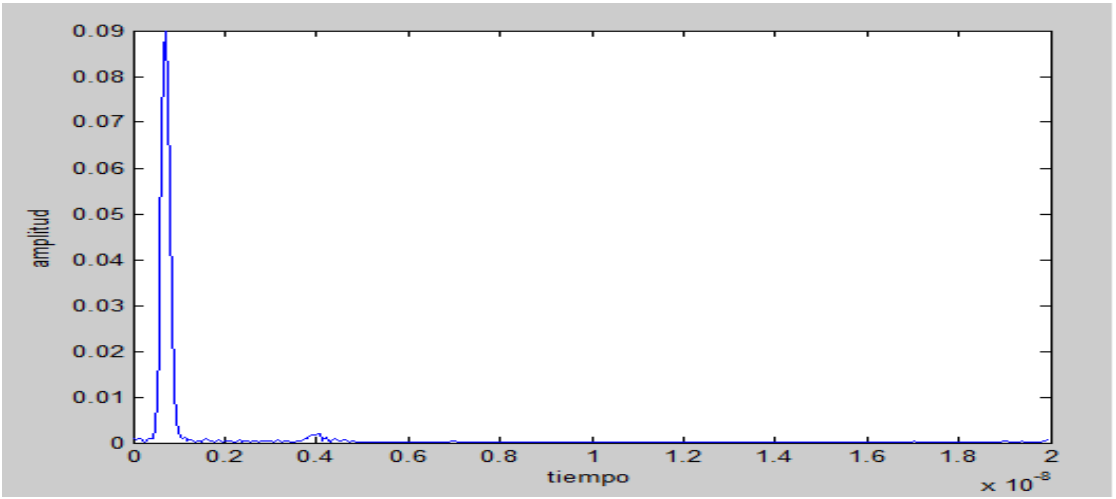

Figura 3.14: La Transformada Discreta de Fourier Inversa de la señal.

4. Análisis de los resultados:

En este apartado se van a analizar los resultados obtenidos con la herramienta desarrollada, así como comparaciones, aclaraciones y soluciones.

4.1 Comparación modo paso banda y modo paso bajo

El modo paso bajo corresponde a la Transformada de Fourier de la señal, por lo que parece que tiene que ser la forma de hacerlo. Sin embargo, el Analizador usa el modo paso banda para señales paso banda. En este apartado, se demuestra el por qué.

En la figura que se muestra a continuación, represento la señal tanto del modo paso banda como del modo paso bajo, en frecuencia:

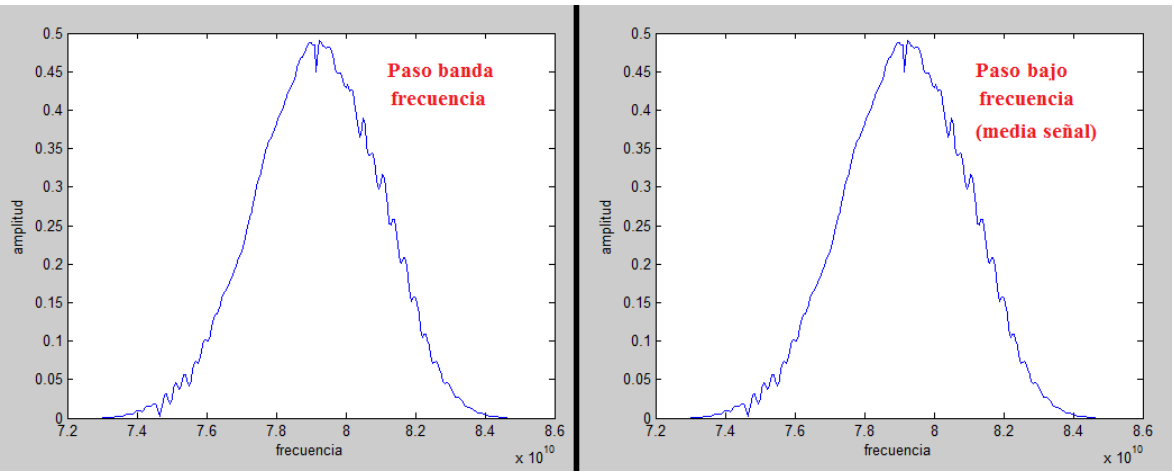

Figura 4.1: Comparación del modo paso banda y del modo paso bajo en el dominio frecuencial.

He querido representar en el modo paso bajo, únicamente media señal, sin la señal conjugada, es decir, la parte negativa de la señal, para que se aprecie que tanto la señal en modo paso banda como en modo paso bajo en frecuencia coinciden, con el pequeño detalle de que solo represento media señal.

A continuación represento la señal completa:

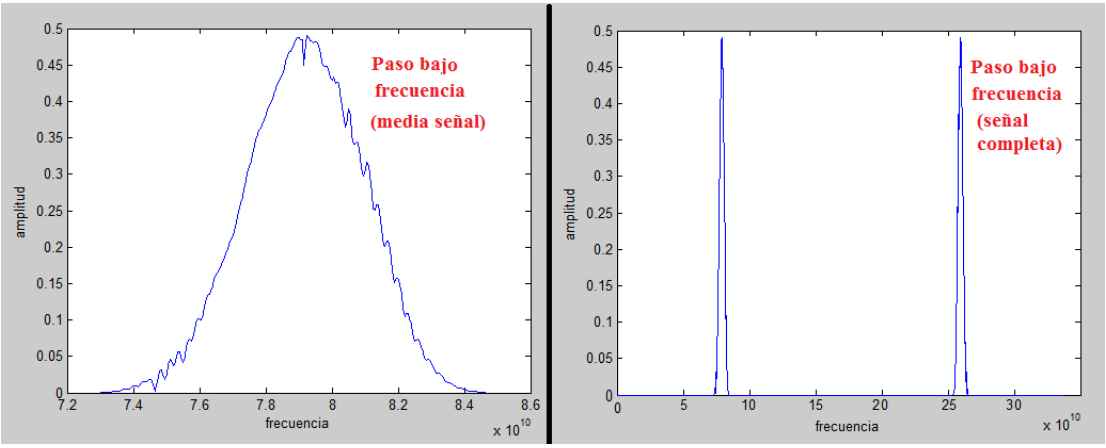

Figura 4.2: Señal en el dominio frecuencial en modo paso bajo.

Como podemos observar, la gran diferencia entre el modo paso banda y el modo paso bajo, es que el modo paso bajo utiliza tanto la parte positiva como la negativa de la señal, a diferencia del modo paso banda que únicamente utiliza la parte positiva de la señal.

Ahora represento la señal en ambos modos en el dominio temporal:

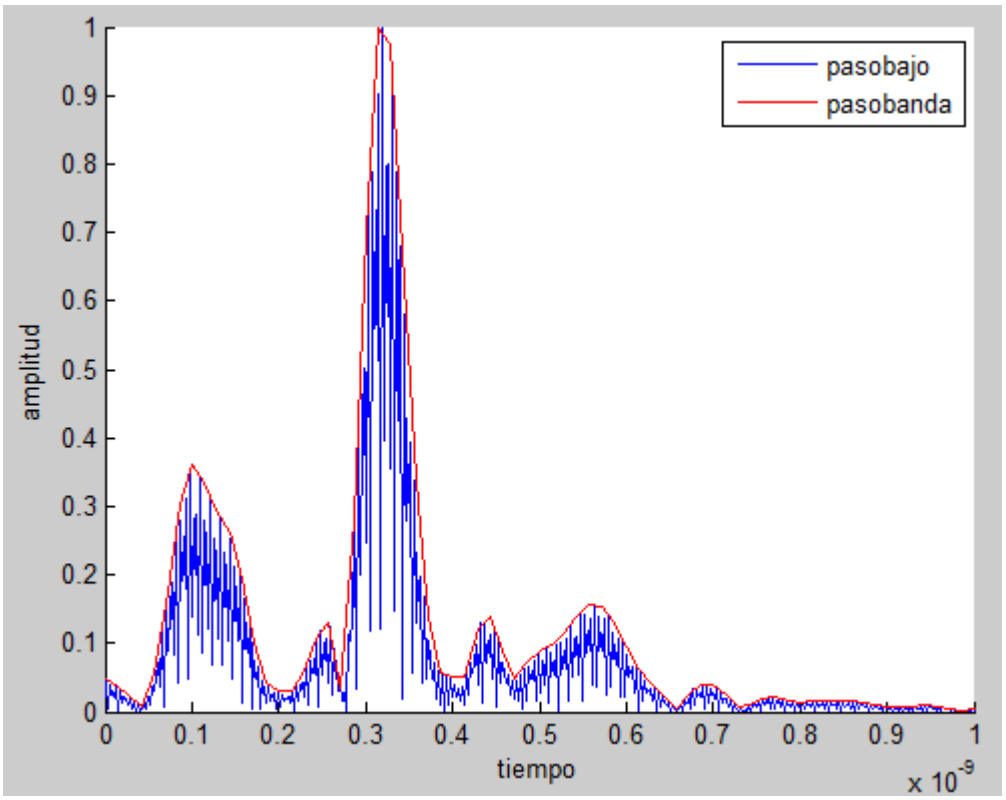

Figura 4.3: Comparación del modo paso banda y del modo paso bajo en el dominio temporal.

Al observar la Figura entendemos claramente por qué el Analizador de redes implementa el modo paso banda y no el paso bajo, ya que resulta difícil distinguir los picos de la respuesta temporal debido al rizado de la señal. La señal del modo paso banda es la envolvente de la paso bajo, distinguiendo claramente los picos de la señal.

4.2 Validación de los resultados

Para comprobar que el programa creado funciona correctamente, se han comparado los resultados obtenidos en el software desarrollado con los resultados obtenidos del Analizador.
Para ello, utilizando el mismo fichero, los comparo tanto en el dominio del tiempo, con gating y sin gating, como en el dominio de la frecuencia.

Se ha trabajado con unos datos provenientes de la medida de un componente en banda W, de 75 a 110 GHz. Su respuesta en frecuencia se muestra en la Figura 4.4:

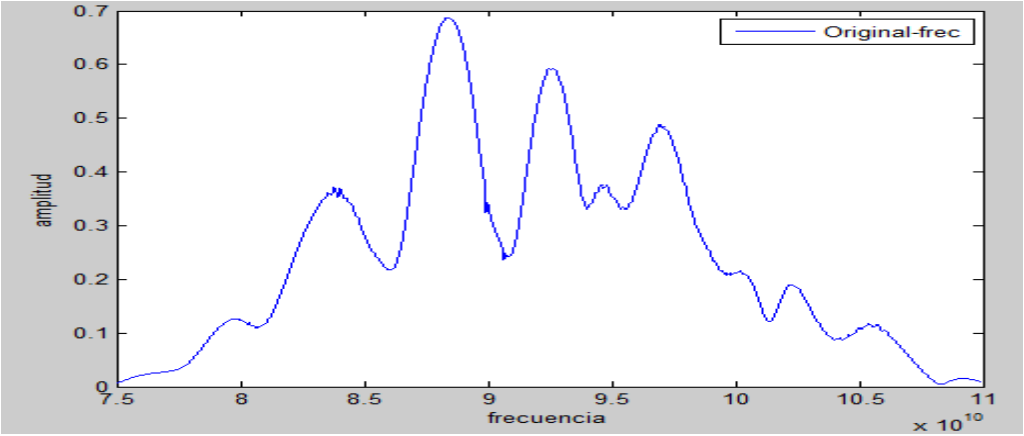

Figura 4.4: Señal en el dominio frecuencial obtenida del Analizador.

A esta señal le aplico la Transformada Discreta de Fourier Inversa para cambiar al dominio temporal. La señal azul es el resultado obtenido al utilizar el software desarrollado y la señal verde es el resultado del Analizador. Esta señal la he calculado en el modo paso banda, tal y como lo hace el Analizador.

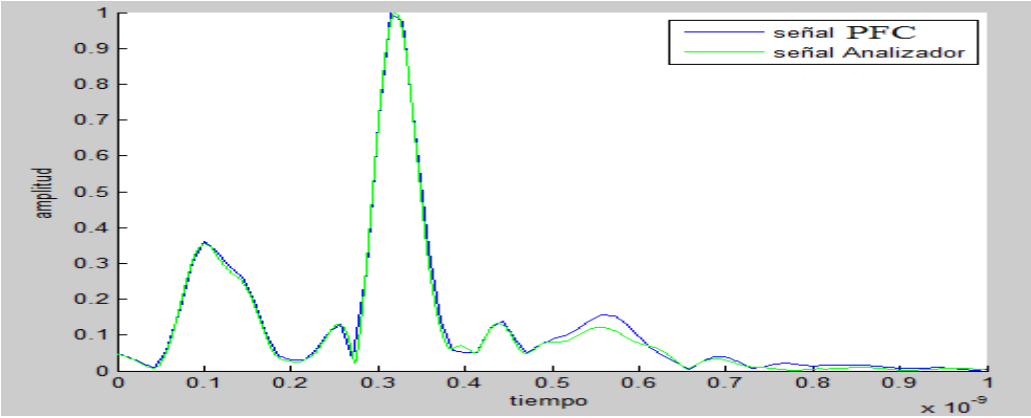

Figura 4.5: Transformada Inversa de la señal obtenida del Analizador y del software desarrollado.

Como podemos observar en la Figura 4.5 ambas señales coinciden, por lo que de aquí llegamos a la conclusión que la Transformada Inversa está bien calculada y, por lo tanto, el cambio del dominio frecuencial al dominio temporal es correcto.

El siguiente paso, es aplicar un *gating temporal*, para ello vamos a tomar los mismos valores que utiliza el Analizador en el fichero ejemplo que tenemos, para aplicar el mismo *gating*.
Los valores que utilizo son 45 picosegundos (10^{-12}) y 185 picosegundos.
Represento el *gating* y recuerdo que la señal azul es el resultado obtenido al utilizar mi software y la señal verde es el resultado del Analizador.

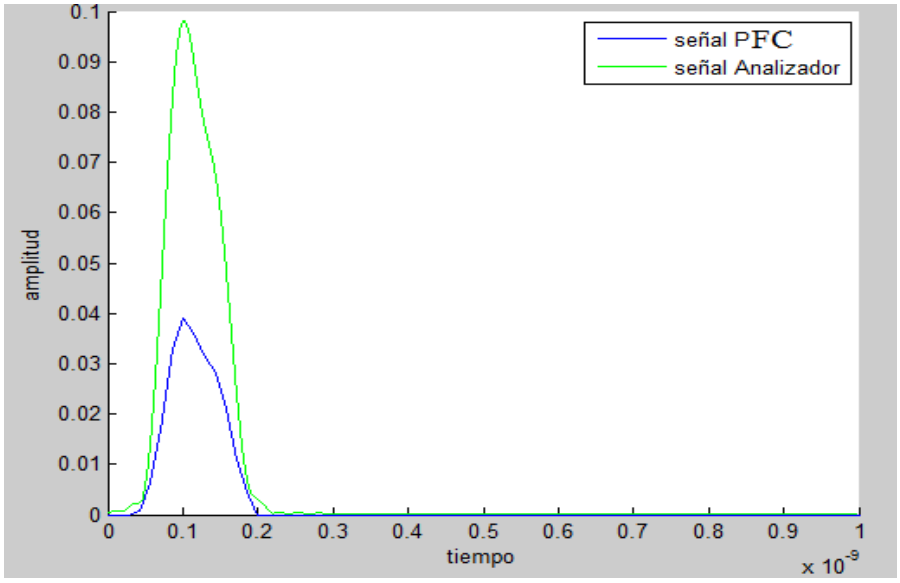

Figura 4.6: Gating temporal de la señal anterior en el dominio temporal entre los valores 45 y 185 ps.

Observamos que hay diferencia entre mi señal con el *gating* aplicado y la del Analizador, puesto que en la señal del Analizador tiene el mismo pico que mi señal, pero con menor amplitud y en los laterales nunca llega a cero, tiene unos lóbulos laterales.

Voy a representarlo también en decibelios para que podamos observarlo mejor:

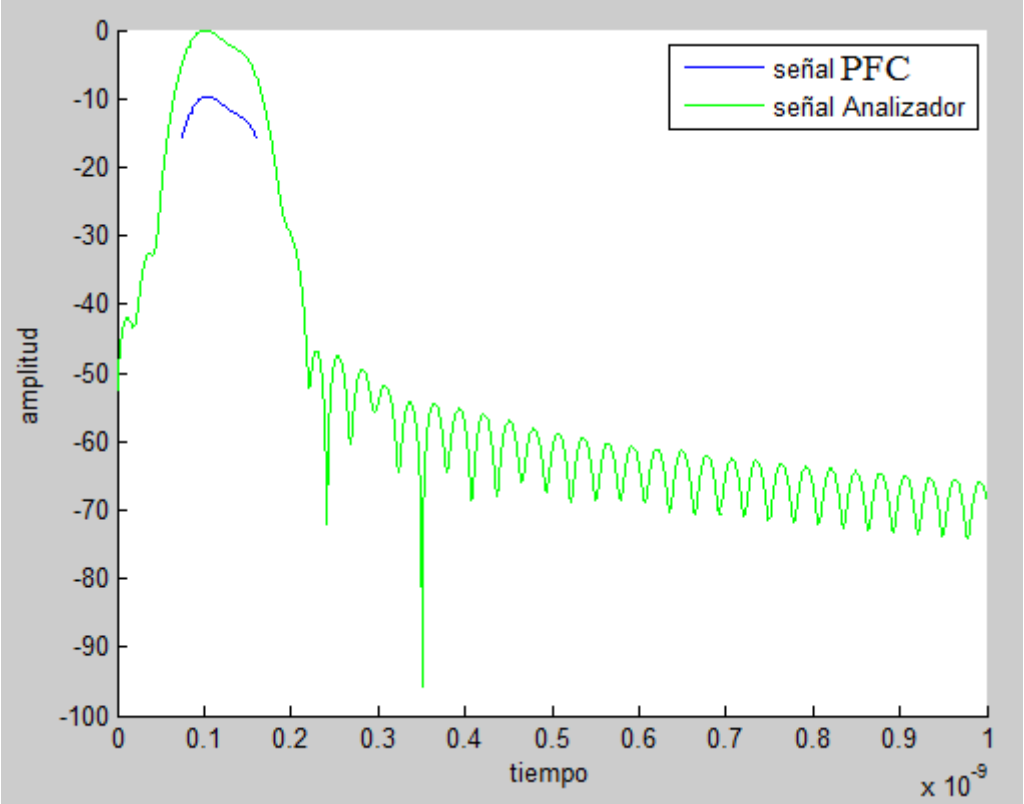

Figura 4.7: Gating temporal entre los valores 45 y 185 ps representado en decibelios.

Al representarlo en decibelios, ya observamos claramente la diferencia entre ambas señales y los lóbulos que posee. Como ambas señales tienen el mismo pico, pero con diferente amplitud, y como la señal del Analizador posee los lóbulos laterales, debido a la renormalización que le aplica.

Otra manera muy clara de verlo, es dividiendo la señal con el *gating* aplicado entre la señal en tiempo sin *gating*, para observar el *gating*:

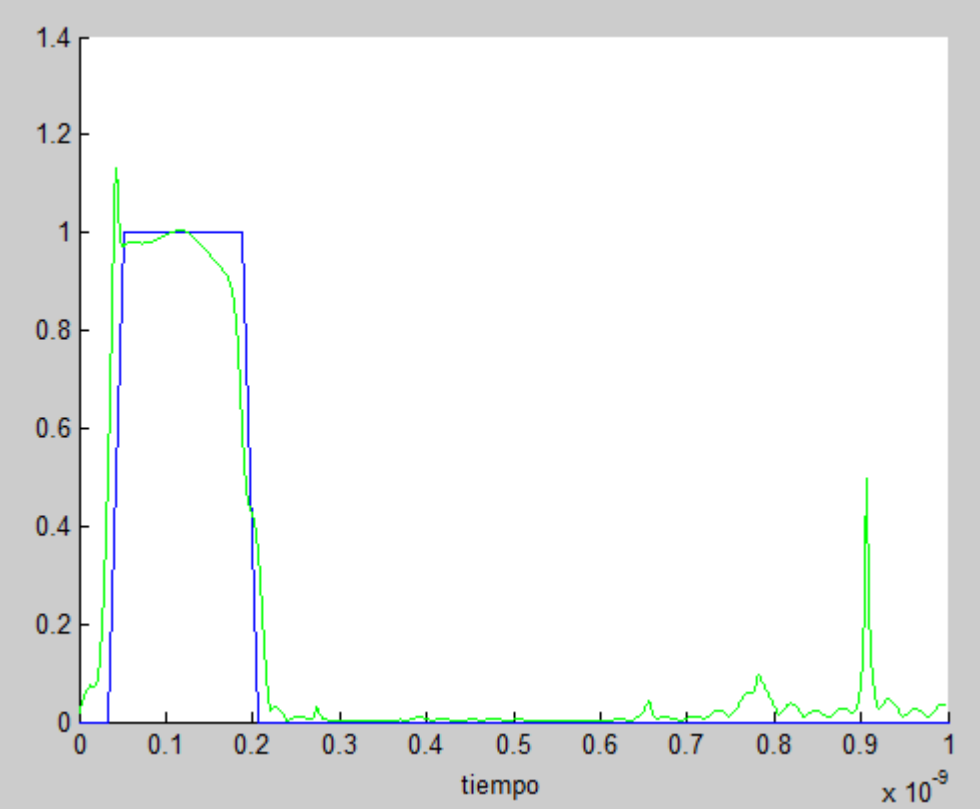

Figura 4.8: Gating aplicado a la señal en tiempo.

De esta manera, se observa claramente el gating que yo le aplico entre 45 y 185 picosegundos, siendo cero todo lo que no se encuentra en este rango y como el Analizador lo aplica de otra manera, no tan sencilla, debido a la normalización que le aplica y que no se especifica en el Manual del Analizador de redes.

A continuación, vamos a calcular la Transformada de Fourier sobre la señal con el *gating* aplicado.

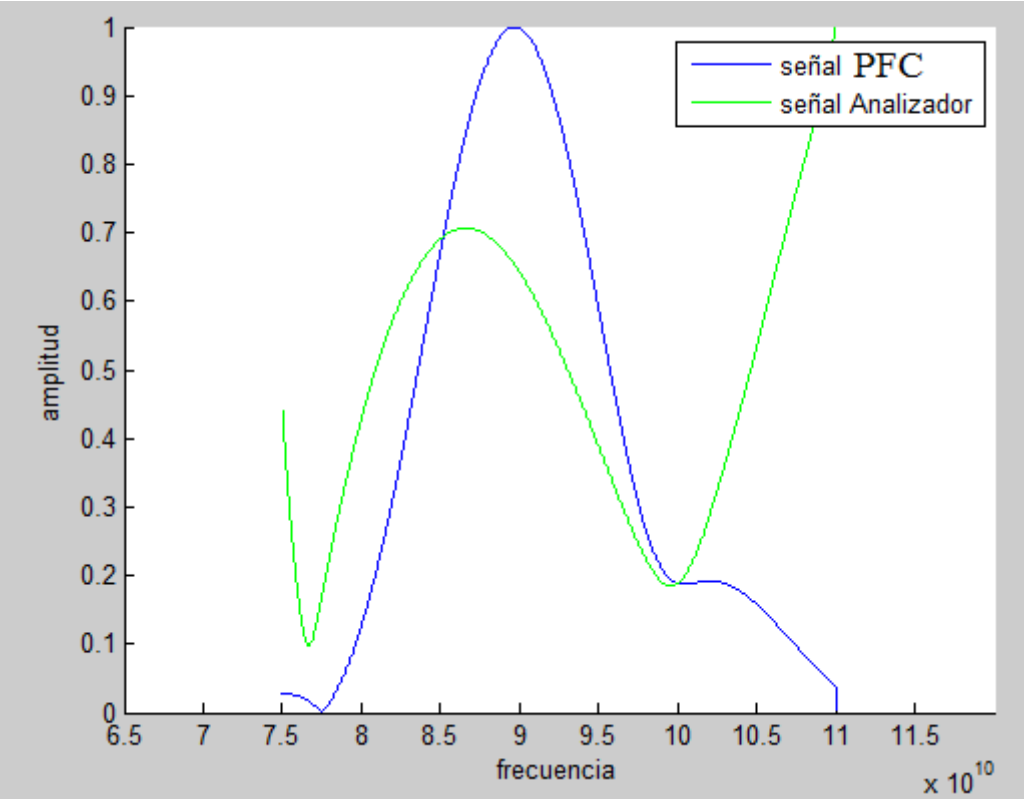

Figura 4.9: Señal en el dominio frecuencial al haberle aplicado la Transformada de Fourier.

Como era de esperar, ambas señales no coinciden en frecuencia, debido a que en el dominio temporal al aplicar el gating, la señal del Analizador no coincide con la mía, al no llegar a ser cero en los laterales y tener esos lóbulos.

Después de todo este desarrollo, llego a las siguientes conclusiones: Mi software obtiene correctamente los datos de los archivos del Analizador, los adapta de forma precisa a éste y los automatiza para cualquier archivo; representa con exactitud la señal en el dominio frecuencial e implementa la Transformada Discreta de Fourier Inversa.

La diferencia entre mi software y el Analizador radica en el *gating temporal*, que como hemos observado en las gráficas anteriores, lo calculo de forma distinta al Analizador de redes vectorial.

Esto se debe a que según el Manual de Agilent: [2] "Time Domain Analysis Using a Network Analyzer" en la página 18 en el apartado 6, una vez aplicado el *gating*, el Analizador realiza una

renormalización de la señal y debido a que en el manual no especifica en qué consiste esa renormalización yo no la he podido implementar en mi software, porque desconozco dichos datos.

4.3 Efecto ventana frecuencial

En la figura que se muestra a continuación observamos una señal sin enventanar y otra enventanada utilizando la ventana de Kaiser-Bessel con α = 3:

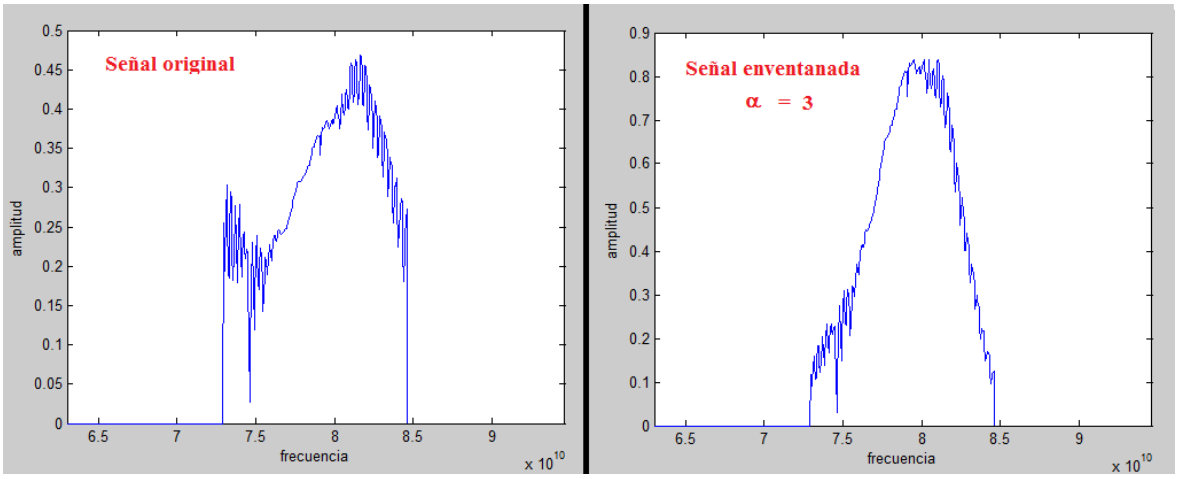

Figura 4.10: Comparación de una señal sin ventana y otra con ventana de Kaiser-Bessel en el dominio frecuencial con α = 3.

La respuesta temporal que obtenemos es la siguiente:

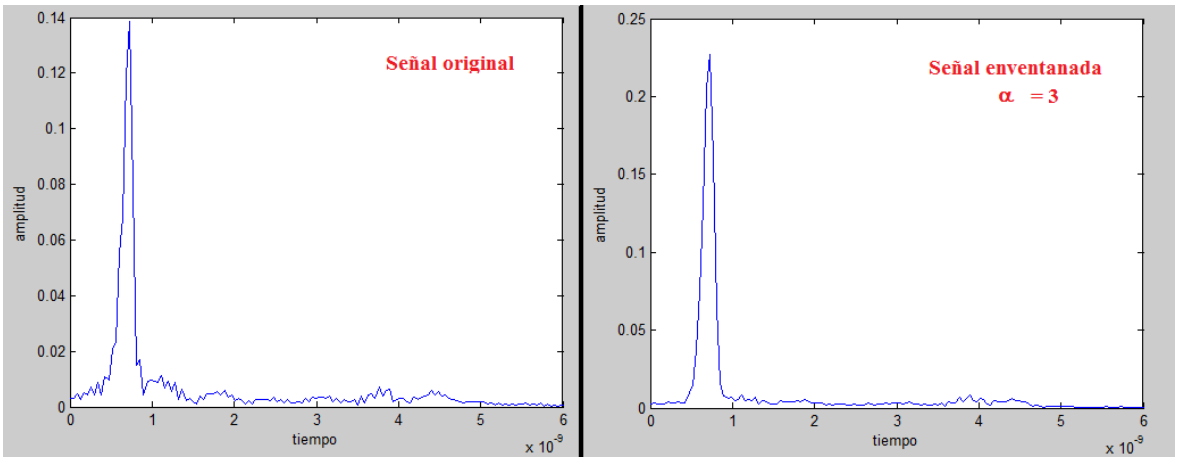

Figura 4.11: Comparación de estas señales en el dominio temporal.

A continuación voy a mostrar dos ejemplos más variando el α de la ventana de Kaiser para observar cómo afecta a la señal.

Utilizando la ventana de Kaiser-Bessel con α = 6:

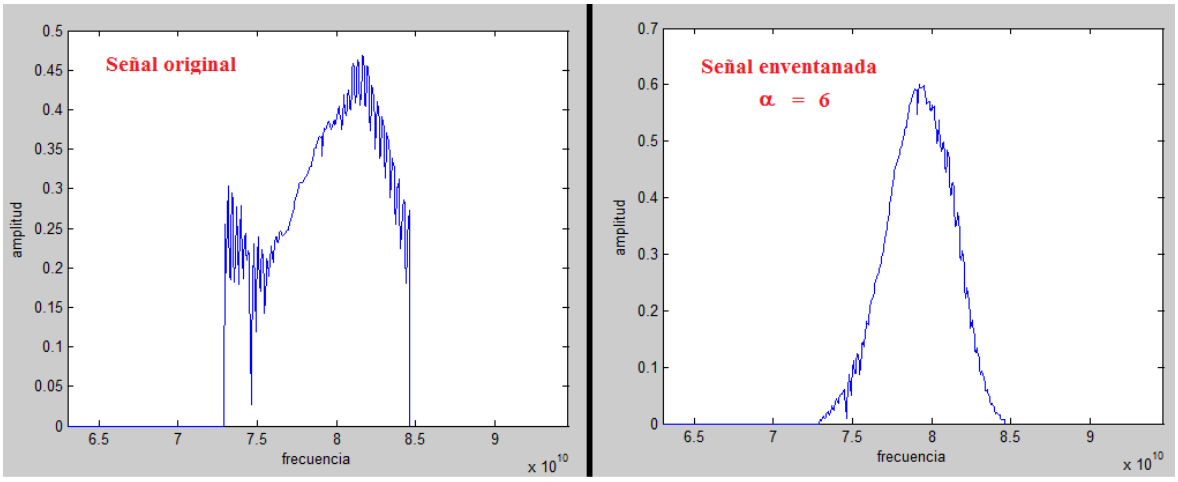

Figura 4.12: Comparación de una señal sin ventana y otra con ventana de Kaiser-Bessel en el dominio frecuencial con α = 6.

La respuesta temporal que obtenemos es la siguiente:

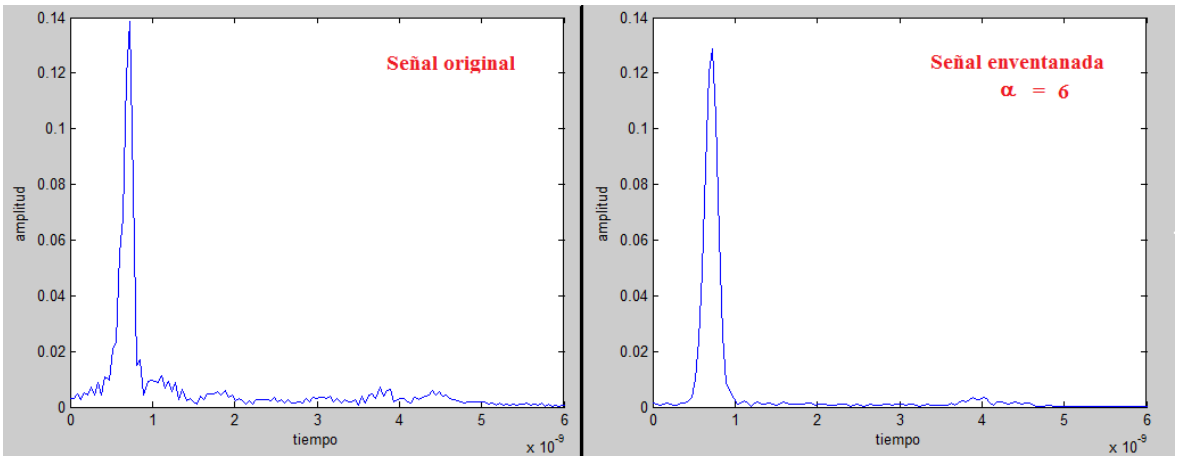

Figura 4.13: Comparación de estas señales en el dominio temporal.

Por último, utilizando la ventana de Kaiser-Bessel con α = 9:

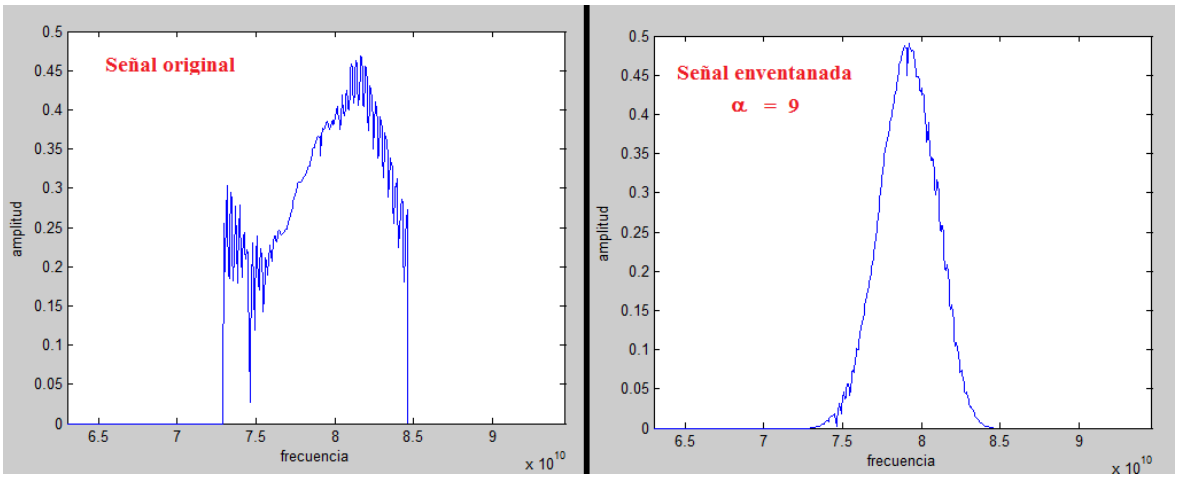

Figura 4.14: Comparación de una señal sin ventana y otra con ventana de Kaiser-Bessel en el dominio frecuencial con α = 9.

La respuesta temporal que obtenemos es la siguiente:

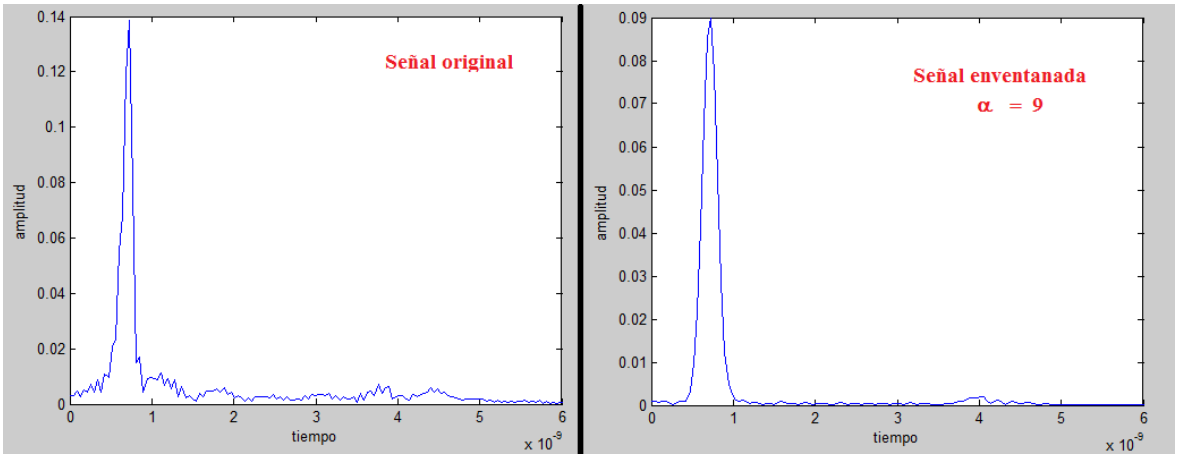

Figura 4.15: Comparación de estas señales en el dominio temporal.

Tal y como hemos observado en estos tres ejemplos, cuanto mayor es el α de la ventana más suaviza la señal en el dominio de la frecuencia y más limpia deja la señal en el dominio temporal. Básicamente, lo que hace esta ventana frecuencial es eliminar las partes menos significativas de la señal para quedarse con la parte más importante, con el pico más grande. Cuanto mayor es α más partes elimina.

El Analizador utiliza esta ventana, ya que da la posibilidad de manejar α, lo cual puede llegar a ser muy útil según la necesidad que tengas, pudiendo aplicar una ventana de α grande para quedarnos con la parte importante de la señal o pudiendo aplicar una ventana de α pequeño para tener mucho detalle de la señal.

4.4 Efecto del *Zero Padding*

El *Zero Padding* es una técnica utilizada habitualmente para aumentar la calidad de representación de la Transformada.

El efecto de esta técnica depende de cuántos ceros introduzcamos desde la interfaz de usuario. Cuantos menos introduzcamos observaremos un cambio menor. La finalidad del *Zero Padding* es incrementar la resolución de las frecuencias, pero ralentiza el cálculo de la Transformada Discreta de Fourier, ya que hay un mayor número de muestras.

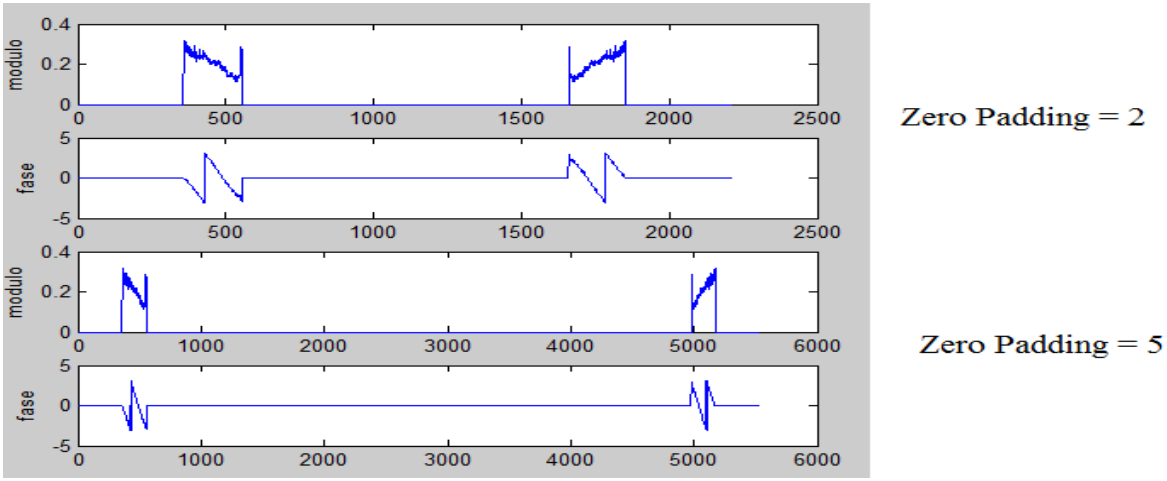

Figura 4.16: Ejemplo de Zero Padding en el dominio de la frecuencia.

Como podemos observar en la figura 4.16, aplicar *Zero Padding* sobre una señal en el dominio de la frecuencia añade ceros en el medio de la secuencia, tanto en el módulo como en la fase. Por lo que aumentar el número de muestras en la secuencia de la frecuencia va a tener como consecuencia una mejora de la representación en el dominio temporal, como vamos a apreciar en la siguiente Figura.

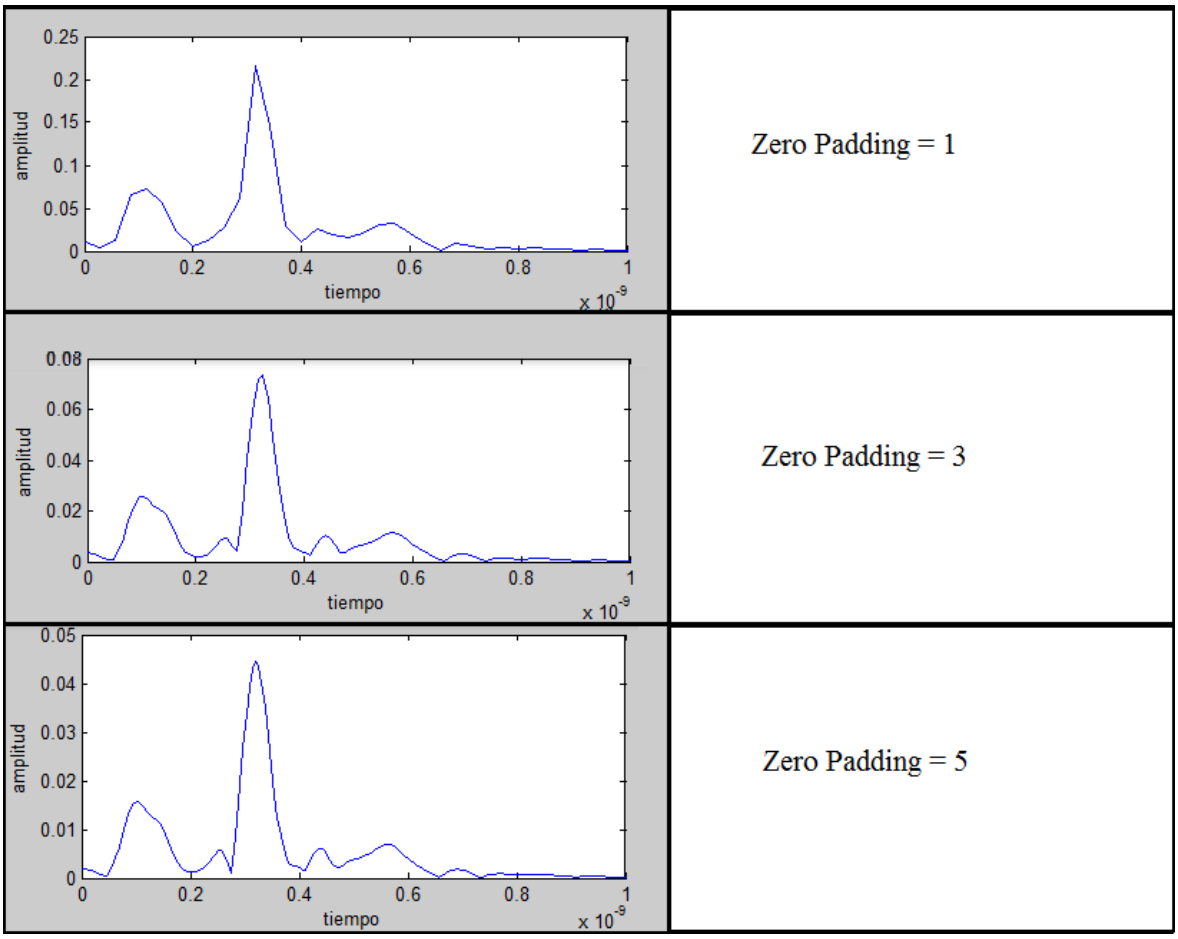

Figura 4.17: Efecto del Zero Padding en la respuesta temporal.

En las Figuras anteriores observamos que si aumentamos el *Zero Padding* en la respuesta temporal, aumentamos el número de muestras y, por tanto, la información que tenemos acerca de la señal. Con el *Zero Padding* igual a cinco tenemos mucha más información acerca de la señal que cuando es igual a uno, esto se debe al número de muestras.

4.5 Efecto de las ventanas temporales

En este apartado voy a analizar los efectos de las ventanas temporales sobre una señal con un *gating* temporal aplicado. La ventana se implementará únicamente sobre la parte de la señal que no sea nula.

4.5.1 Efecto de la ventana de Hamming

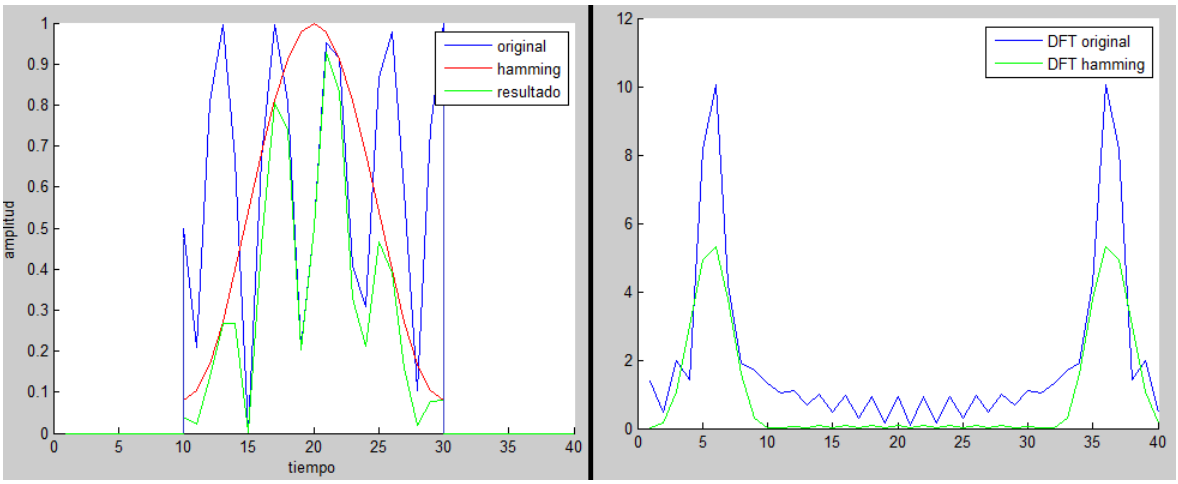

Figura 4.18: Comparación implementando una ventana de Hamming a una señal y su Transformada.

De la gráfica se puede interpretar que en el dominio temporal tiene un lóbulo principal muy ancho, pero los laterales no llegan a ser cero, por lo que si observamos la gráfica en el dominio frecuencial, vemos que hay un rizado rondando el valor cero que no se llega eliminar, pero a su vez, gracias al lóbulo principal obtenemos unos buenos picos en el espectro.

4.5.2 Efecto de la ventana Triangular

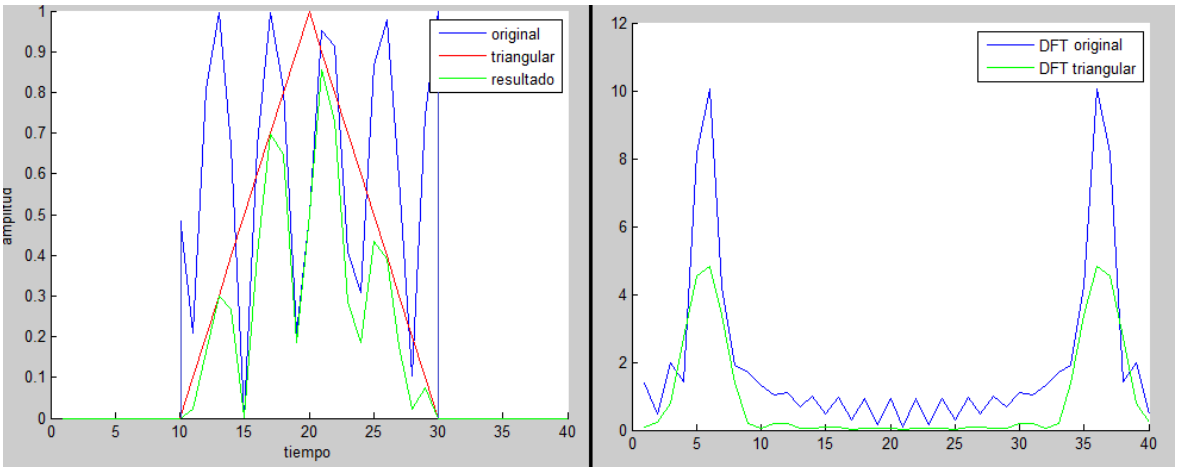

Figura 4.19: Comparación implementando una ventana de Triangular a una señal y su Transformada.

Se obtiene un pico bastante pronunciado en el espectro, aunque menor que con Hamming, y suavizando correctamente los lóbulos laterales, de tal modo que son prácticamente nulos en el espectro. El resultado es bastante similar que con Hamming.

4.5.3 Efecto de la ventana de Hanning

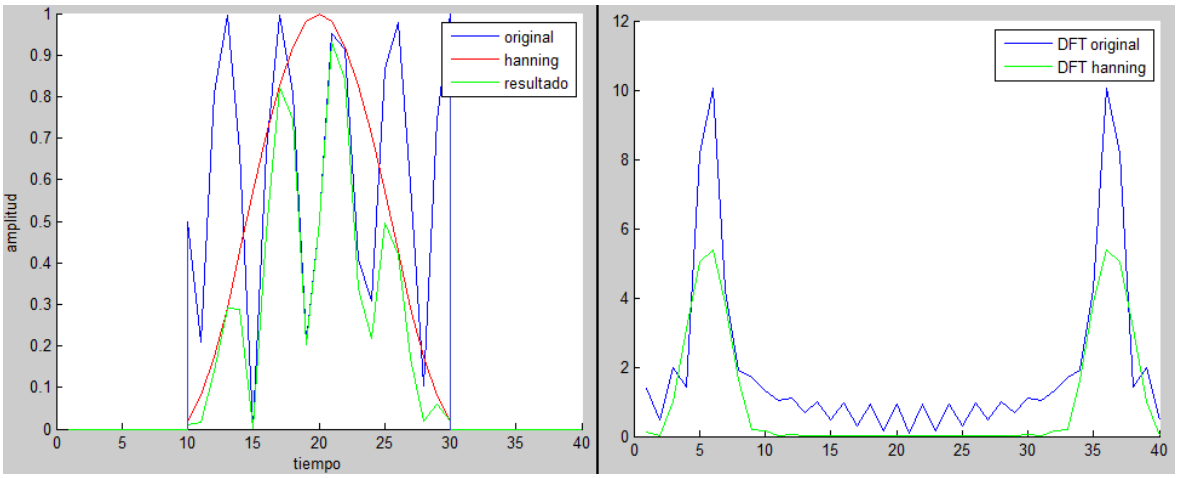

Figura 4.20: Comparación implementando una ventana de Hanning a una señal y su Transformada.

Lóbulo principal muy ancho, obteniendo dos picos muy destacados. Elimina por completo los lóbulos laterales, tal y como se observa en el espectro.
Ventana muy similar a la de Hamming con la clara diferencia de los lóbulos laterales.

4.5.4 Efecto de la ventana de Blackman

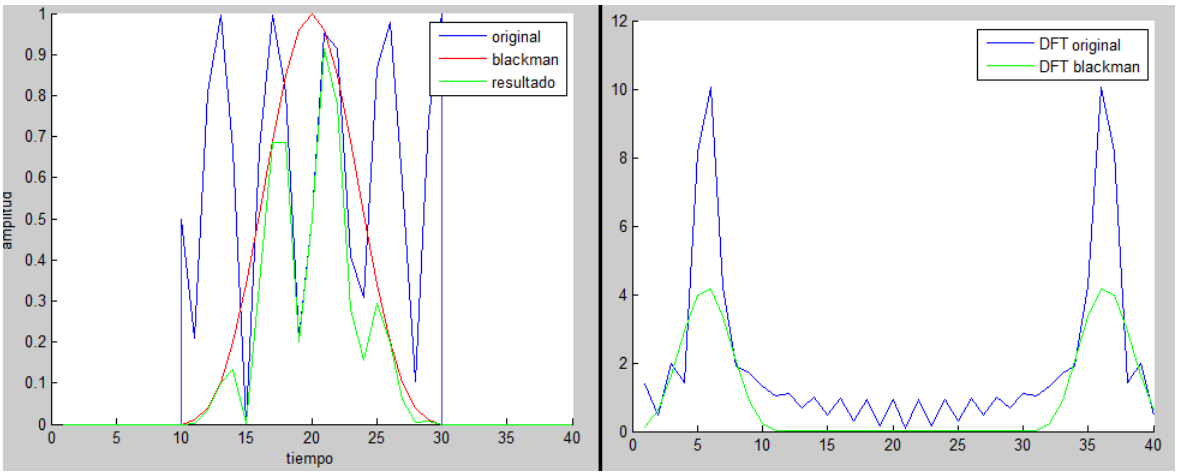

Figura 4.21: Comparación implementando una ventana de Blackman a una señal y su Transformada.

Esta ventana posee un lóbulo principal más estrecho, por eso que en el espectro los picos no estén tan pronunciados como en las otras ventanas, pero al igual que con la ventana de Hanning, elimina los lóbulos laterales en el espectro.

5. Conclusiones y Líneas futuras

En el presente Proyecto Final de Carrera (PFC) he implementado un software para el procesado de los datos obtenidos mediante un analizador vectorial de redes en el dominio del tiempo, incluyendo herramientas de postprocesado.

El software implementado puede ejecutarse en cualquier versión del software Matlab, ya que lo he programado utilizando comandos universales, comunes a todas las versiones. Se ha comprobado su correcto funcionamiento en varias versiones: R2009b y R2012b.

Este software analiza los datos de los archivos obtenidos del Analizador, obtiene el módulo y la fase de la señal en el dominio frecuencial, se le puede aplicar *Zero Padding* y una ventana frecuencial (Kaiser-Bessel) y cambia del dominio frecuencial al temporal utilizando la Transforma Discreta de Fourier Inversa. Además de eso, dispone de herramientas de postprocesado en el dominio temporal como el *gating* y diversas ventanas.

Se ha conseguido que el software obtenga los datos de los archivos provenientes del Analizador y los adapte a éste, que era el primer objetivo del proyecto. No importa el número de datos que haya que analizar, puesto que está capacitado para cualquier número.

El segundo objetivo era implementar la Transformada Discreta de Fourier Inversa (DFTI) sobre la señal. Para ello, lo primero es definir el modo en el que deseemos implementar el software, en modo paso banda o en modo paso bajo. Una vez definido el modo, se construye la señal y a esta se le implementa la DFTI. En el apartado de Validación de resultados ya demostramos que está logrado el objetivo.

El tercer objetivo era implementar las herramientas de postprocesado. Se han implementado las herramientas de gating y diferentes ventanas, todo ello en el dominio temporal. En la presente Memoria se ha comprobado que las ventanas temporales funcionan correctamente.
El *gating* temporal no se ha conseguido, puesto que el Analizador aplica una renormalización y no es tan simple como se ha calculado.

Por último, el último objetivo era crear una interfaz gráfica. El objetivo se ha logrado, puesto que ha quedado una interfaz muy completa e intuitiva, de manera que de forma sencilla se puede analizar una señal tanto en tiempo como en frecuencia.

Como líneas futuras, queda por aplicar una renormalización del *gating* temporal, ya que no se ha conseguido esa información por no especificarse en el Manual del Analizador.
Una vez se haya descubierto esa información, no es más que aplicar dicha renormalización al *gating* y habremos conseguido el mismo resultado que el Analizador vectorial de redes.

6. Manual de usuario

El procedimiento para el procesado de las señales provenientes del analizador de redes vectorial desde la interfaz de usuario es el siguiente:

(Para seguir el procedimiento adecuadamente, ver figura 48 en la página siguiente)

Lo primero es seleccionar el archivo a procesar (botón 8) que se mostrará en el texto (9) y podremos comprobar si hemos seleccionado el que deseamos o no. A continuación, introducimos un valor de *Zero Padding* (12) y un valor de Ventana Kaiser-Bassel (13), para poder visualizar la señal según nuestras necesidades. También, seleccionamos la opción del modo que deseemos (14), paso bajo o paso banda. Lo siguiente es pulsar el botón Ejecutar (10). Como podemos observar, nos ha aparecido el módulo de la señal en la gráfica 5, la fase de la señal en la gráfica 6 y la Transformada Discreta de Fourier Inversa de dicha señal en la gráfica 7.

A continuación, podemos procesar la señal mediante la utilización de herramientas adicionales. Podemos aplicar un *gating temporal* a la Transformada Discreta de Fourier Inversa de la señal, introduciendo los valores de tiempo, t1 (15) y t2 (16) y pulsando el botón gating (17), obteniendo en la gráfica 23 el resultado. Además, podemos aplicar distintos tipos de ventanas al *gating temporal* calculado anteriormente, simplemente pulsando en el botón de la ventana que prefiramos. Las ventanas que podemos elegir son: Hamming (18), Triangular (19), Hanning (20), Blackman (21) y Rectangular (22). Al pulsar el botón de cada ventana, mostraremos la ventana calculada en la gráfica 23 y la Transformada Discreta de Fourier de dicha señal enventanada en la gráfica 24.

Por último, nombrar unas opciones muy útiles, como la de resetear (11), que como su nombre bien indica, resetea todas las gráficas y datos para posteriormente poder analizar otras señales. También tenemos otra opción, que es la de guardar los datos automáticamente, en caso de seleccionar la opción (25). Guarda los datos de la Transformada Discreta de Fourier Inversa, de la Transformada Discreta de Fourier de cada ventana y la señal en tiempo de cada ventana en un archivo que se guardará en nuestra carpeta de trabajo en formato .csv de la señal representada en las gráficas 7 (datos de la Transformada Discreta de Fourier Inversa) y 24 (datos de la Transformada Discreta de Fourier). Además de los datos, nos guarda la gráfica en un archivo que se guardará en

nuestra carpeta de trabajo en formato .jpg, correspondiente a la Transformada Discreta de Fourier Inversa y/o a la Transformada Discreta de Fourier, de sus respectivas gráficas.

Las últimas cuatro opciones, son la del cursor (1), *zoom* in (2), *zoom* out (3) e imprimir (4). Estas cuatro opciones, nos sirven para que la interfaz sea más accesible y podamos interactuar más fácilmente, pudiendo acercarnos, alejarnos, movernos por las gráficas e imprimir la interfaz de usuario.

Leyenda	
1	Cursor
2	Zoom in (acercar)
3	Zoom out (alejar)
4	Imprimir
5	Gráfica que muestra el módulo de la señal
6	Gráfica que muestra la fase de la señal
7	Gráfica que muestra la dft inversa de la señal
8	Seleccione el archivo a analizar
9	Muestra el archivo seleccionado
10	Ejecuta el archivo a analizar
11	Resetea todos los datos y gráficas
12	Introducir Zero Padding
13	Introducir tamaño ventana frecuencial
14	Seleccione el modo
15	Introducir el t1 del gating temporal
16	Introducir el t2 del gating temporal
17	Ejecutar el gating temporal
18	Aplicar la ventana de Hamming
19	Aplicar la ventana Triangular
20	Aplicar la ventana de Hanning
21	Aplicar la ventana de Blackman
22	Aplicar la ventana Rectangular
23	Gráfica que muestra el gating y las ventanas
24	Gráfica que muestra la DFT
25	Guardar Datos Automaticamente

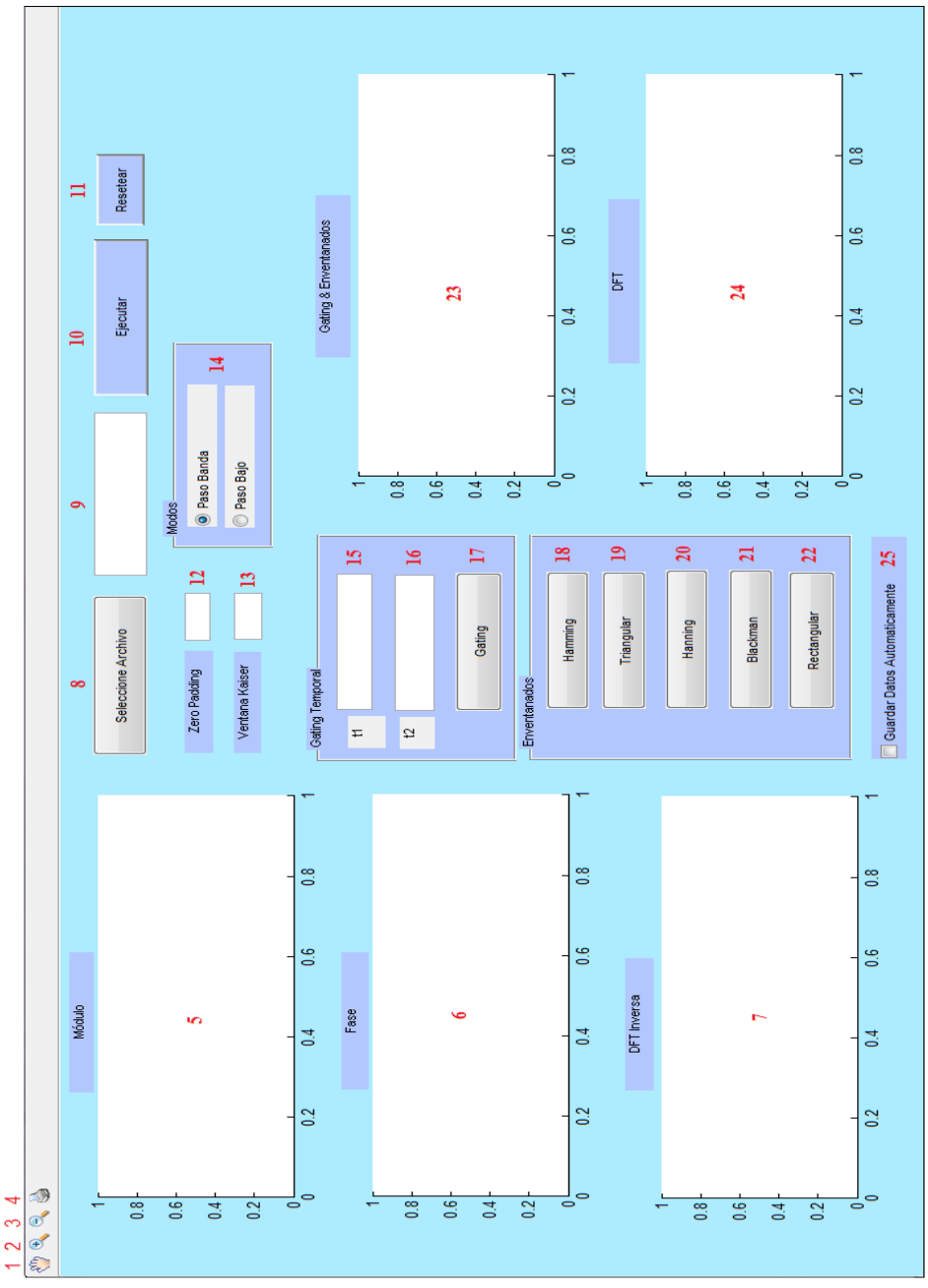

Figura 6.1: Esquema de la interfaz gráfica del proyecto. Manual de usuario.

7. Bibliografía

[1] “Matlab: An Introduction with Applications”, John Wiley & Sons, Inc.

[2] “Time Domain Analysis Using a Network Analyzer”, Agilent Application Note 1287-12.

[3] J. Dunsmore, “Gating Effects in Time Domain Transforms”, ARFTG Microwave Measurement Symposium, 2008 72nd, 9-12 Dec. 2008.

[4] Time Domain Measurements using Vector Network Analyzer ZVR, Rohde & Schwartz Application Note 1EZ44_0E.

[5] “Matlab: Creating Graphical User Interfaces”, John Wiley & Sons, Inc.

[6] MathWorks http://www.mathworks.es/

[7] “Procesado digital de señales: Fundamentos para comunicaciones y control”, Eduard Beltrán Albertí.

[8] A.V. Oppenheim, R.W. Schafer, “Tratamiento de Señales en Tiempo Discreto”, Prentice Hall.

8. Apéndice 1: Explicación del código del proyecto

8.1 Cabecera de la interfaz gráfica

```
function varargout = interfaz(varargin)
% INTERFAZ M-file for interfaz.fig
%      INTERFAZ, by itself, creates a new INTERFAZ or raises the existing
%      singleton*.
%
%      H = INTERFAZ returns the handle to a new INTERFAZ or the handle to
%      the existing singleton*.
%
%      INTERFAZ('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in INTERFAZ.M with the given input arguments.
%
%      INTERFAZ('Property','Value',...) creates a new INTERFAZ or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before interfaz_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to interfaz_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help interfaz

% Last Modified by GUIDE v2.5 21-Mar-2013 19:53:32

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @interfaz_OpeningFcn, ...
                   'gui_OutputFcn',  @interfaz_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
```

Figura 8.1: Código de Matlab de la cabecera de la interfaz.

Este es el código que se crea por defecto con Matlab al crear una interfaz de usuario, con el nombre de la interfaz en rojo, en mi caso llamado 'interfaz.m'. Este código no hay que modificarlo.

8.2 Inicialización de la interfaz

```
function interfaz_OpeningFcn(hObject, ~, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to interfaz (see VARARGIN)

cla(handles.modulo_axes,'reset');%inicializo las graficas
cla(handles.fase_axes,'reset');
cla(handles.dftinversa_axes,'reset');
cla(handles.ventana_axes,'reset');
cla(handles.dft_axes,'reset');

% Choose default command line output for interfaz
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
```

Figura 8.2: Código de Matlab de otra parte de la cabecera de la interfaz.

Este código también es parte de la cabecera, sólo que en esta función se ponen las variables que queremos inicializar o crear, al abrir la interfaz.

En mi caso, inicializo todas las gráficas en blanco.

Lo que viene a continuación ya es el código de mi programa.

8.3 Selección de archivo

```
%%%%%%%%%% SELECCIONO ARCHIVO %%%%%%%%%%
function pbSelect_Callback(~, ~, handles)
% hObject    handle to pbSelect (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

fileName=uigetfile('*.csv');%elijo y leo el archivo .csv
set(handles.editFileName,'string',fileName)%escribo el nombre del archivo en el recuadro
```

Figura 8.3: Código de Matlab del botón Seleccionar Archivo.

Con el comando uigetfile se me abre una ventana para seleccionar el archivo .csv que deseemos abrir. Con el comando set escribimos el nombre del archivo seleccionado en el recuadro que se encuentra justo al lado del botón de selección de archivo, ya que el comando handles, nos indica el objeto donde lo deseamos escribir.

8.4 Cálculo del número de filas del archivo seleccionado

```
fid=fopen(fileName);%abro el archivo
temp=fgets(fid);%leo la primera fila
cont=1;%inicializo contador a 1
tf=0;%inicializo tf a 0
while tf~=1%mientras tf sea distinto de 1
    tf=strncmpi(temp,'END',3);%comparo 3 digitos con
    %la palabra 'END'. Si lo encuentro tf=1, sino tf=0
    if tf==1%si tf es igual a 1
       fclose(fid);%salgo del bucle
        break
    else%si tf es igual a 0
    temp=fgets(fid);%leo la siguiente fila
    cont=cont+1; %incremento en 1 el contador
    end
end
cont=cont-9;%resto las filas de la cabecera
```

Figura 8.4: Código correspondiente al cálculo del número de filas del archivo seleccionado.

En este código calculo el número de filas que tiene el archivo seleccionado para poder leer correctamente el archivo, ya que dicho archivo no tiene únicamente los datos, sino que además incluye información acerca del analizador, llamada cabecera, que no me interesa, y con esta función consigo seleccionar solo la parte que deseo. El número de filas lo asigno a la variable "cont".

8.5 Lectura, extracción y adaptación de los datos del archivo seleccionado

```
medida=csvread(fileName,8,0,[8 0 cont 2]);%abro el archivo a analizar
fprimera=medida(1,1);%leo la primera frecuencia
fsegunda=medida(2,1);%leo la segunda frecuencia
fultima=medida(a,1);%leo la ultima frecuencia
frecuencias=medida(1:a,1);%leo todas las frecuencias

AdB=medida(1:a,2);%leo todos los modulos de la señal
A=10.^(AdB/20);%paso de dB a lineal

FaseGrados=medida(1:a,3);%leo todas las fases de la señal
Fase=(FaseGrados.*pi)/180;%paso de grados a radianes

diff1f2=fsegunda-fprimera;%calculo la diferencia entre f1 y f2
k1=round(fprimera/diff1f2);%mediante una regla de 3 calculo k1 en funcion de f1 y de la diferencia
k2=round(fultima/diff1f2);%hago lo mismo con k2, solo que con la ultima frecuencia
```

Figura 8.5: Código de Matlab de la lectura, extracción y adaptación de los datos.

La parte dibujada en verde de la Figura 8.6 corresponde a la lectura del archivo .csv utilizando el comando csvread. Los datos entre paréntesis son los que indican qué archivo queremos

leer y a partir de qué fila y columna queremos leer los datos. A dicho archivo le asigno la variable "medida".

La parte dibujada en azul de la Figura 30 corresponde a la extracción de datos del archivo. El primer valor del paréntesis es la columna y el segundo la fila.

La parte roja corresponde a la adaptación de los datos del archivo al software, explicado anteriormente en el apartado 4. Recopilación y presentación de datos.

8.6 *Zero Padding*

```
zero=str2num(get(handles.edit_zero,'string'));%zeropadding
N=k2*zero;
```

Figura 8.6: Código correspondiente al cálculo del Zero Padding.

Con este código leo el dato introducido desde la interfaz con el comando get y lo transformo de string a número con el comando str2num. A continuación, lo multiplico por la última frecuencia y obtengo el número de muestras.

8.7 Construcción de la señal en modo paso bajo

```
v=zeros(1,N);%creo un vector de ceros de tamaño N.
v(k1:k2)=A.*exp(1j.*Fase);%construyo la señal
L=2*N;%creo la variable L que es el doble del numero de muestras
xnorm=zeros(1,N);%inicializo a ceros
xconj=zeros(N+1,L);%inicializo a ceros
x=zeros(1,L);%inicializo a ceros
xnorm=v;
conjugado=conj(xnorm);%calculo la señal conjugada
xconj=flipdim(conjugado,2);%doy la vuelta al vector conjugado de xnorm
x=[xnorm xconj];%concateno la señal
```

Figura 8.7: Código correspondiente a la construcción de la señal.

Con este código se construye la señal completa. Asigno a la variable "v" la parte "positiva" de la señal, tal y como lo explico en el apartado 4. Recopilación y representación de datos. Posteriormente, calculo la señal conjugada, la parte "negativa" de la señal, utilizando el comando conj. Por último, concateno ambas partes y obtengo la señal.

8.8 Cálculo de la Transformada Discreta de Fourier Inversa

```
dft=ifft(x);%calculo la dft inversa de la señal
```

Figura 8.8: Código del cálculo de la Transformada Discreta de Fourier Inversa.

Utilizo el comando ifft para calcular la Transformada Discreta de Fourier Inversa.

8.9 Representación de la señal y de su Transformada Discreta de Fourier Inversa

```
fs=2*diff1f2*N;%calculo la frecuencia de muestreo
t=(1:L)/fs;%t=n/fs calculo el tiempo en función del nº muestras y frecuencia de muestreo

axes(handles.modulo_axes)%represento el modulo
plot(frecuencias,A)
xlabel('frecuencia')
ylabel('amplitud')
axes(handles.fase_axes)%represento la fase
plot(frecuencias,Fase)
xlabel('frecuencia')
ylabel('amplitud')
axes(handles.dftinversa_axes)%represento la dft inversa
plot(t,abs(dft))
xlabel('tiempo')
ylabel('amplitud')
```

Figura 8.9: Código asociado a la representación de la señal, en módulo y fase, y de la Transformada Discreta de Fourier Inversa.

Represento la señal en módulo y fase, junto con su Transformada Discreta de Fourier Inversa, para ello utilizo el comando plot para representar, axes y handles para especificar en qué gráfica representarlo y xlabel e ylabel para nombrar los ejes.

8.10 Checkbox

```
if (get(handles.checkbox,'Value') == get(handles.checkbox,'Max'))% El checkbox ha sido marcado

    matriz=[t' dft'];%creo una matriz con los datos dft y t
    csvwrite('datos_dftinversa.csv',matriz);%escribo los datos en un archivo

    f=getframe(handles.dftinversa_axes,[-36 -18 450 210]);
    im=frame2im(f);
    imwrite(im,'grafica_dftinversa.jpg');%guardo la gráfica de la dft inversa
end
```

Figura 8.10: Código perteneciente al Checkbox.

La estructura del Checkbox requiere de una condición, tal y como vemos en la Figura 8.11, en este caso, si el Checkbox está seleccionado me guardará los datos, en caso de no estarlo no los

guardará. Para guardar los datos utilizo el comando csvwrite y para guardar las gráficas utilizo el comando imwrite.

8.11 Gating Temporal

```
t1=str2num(get(handles.t1_input,'string'));%valores introducidos desde interfaz
t2=str2num(get(handles.t2_input,'string'));
[min1,ind1]=min(abs(t-t1));%indice de t1
[min2,ind2]=min(abs(t-t2));%indice de t2

gating=zeros(1,L);
gating(ind1:ind2)=dft(ind1:ind2);
dft1=fft(gating)';
```

Figura 8.11: Código del gating temporal.

Asigno "t1" y "t2" al intervalo en el cual aplicamos el *gating temporal*, para ello, con el comando get cojo el valor introducido desde pantalla y con el comando str2num lo paso de string a número. Debido a que "t1" y "t2" no son números enteros, calculo el índice de estos valores dentro del vector de t (tiempo).

Una vez tengo ambos valores, aplico el *gating*, siendo la señal ceros exceptuando el rango de valores entre "t1" y "t2".

A este rango de valores es al que implemento la Transformada de Fourier para obtener la señal en frecuencia.

8.12 Botón reiniciar

```
%%%%%%%%%% BOTON DE REINICIAR %%%%%%%%%%
function pbreiniciar_Callback(~, ~, handles)
% hObject    handle to pbreiniciar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
reinicio='';
set(handles.editFileName,'string',reinicio);%reinicio todas las graficas y textos
set(handles.t1_input,'string',reinicio);
set(handles.t2_input,'string',reinicio);
set(handles.edit_zero,'string',reinicio);
cla(handles.modulo_axes,'reset');
cla(handles.fase_axes,'reset');
cla(handles.dftinversa_axes,'reset');
cla(handles.ventana_axes,'reset');
cla(handles.dft_axes,'reset');
```

Figura 8.12: Código del Botón Reiniciar.

Utilizo el comando cla seguido de la gráfica o texto que quiero reiniciar.

8.13 Tipos de ventanas

8.13.1 Ventana temporal de Hamming

```
gating=zeros(1,L);
gating(ind1:ind2)=dft(ind1:ind2);

xw1=zeros(1,L);
w1=hamming(L);%ventana de hamming
xw1=gating.*w1';%aplico la ventana a la señal

axes(handles.ventana_axes)%represento la ventana
plot(t,abs(xw1))
legend('Ventana Hamming')
xlim([t1 t2])
xlabel('tiempo')
ylabel('amplitud')

derrame1=fft(xw1);%calculo la dft de la ventana
tam=length(derrame1)/2;
dft1=derrame1(1:tam);%represento solo media señal

axes(handles.dft_axes)%represento la dft
plot(abs(dft1))
legend('DFT Ventana Hamming')
xlabel('frecuencia')
ylabel('amplitud')
```

Figura 8.13: Cálculo de la ventana de Hamming junto con su correspondiente Transformada Discreta de Fourier y sus representaciones.

Como podemos observar en la Figura 8.14, implemento la ventana de Hamming, utilizando el comando haming, sobre la señal con el *gating temporal* aplicado y la represento en su correspondiente gráfica. Posteriormente calculo su Transformada Discreta de Fourier, utilizando el comando fft, y la represento.

Sólo represento media señal, porque es la parte que me interesa, la otra parte es exactamente igual sólo que invertida.

8.13.2 Ventana temporal Triangular

```
gating=zeros(1,L);
gating(ind1:ind2)=dft(ind1:ind2);

xw2=zeros(1,L);
w2=bartlett(L);%ventana de bartlett
xw2=gating.*w2';%aplico la ventana a la señal

axes(handles.ventana_axes)%represento la ventana
plot(t,abs(xw2))
legend('Ventana Triangular')
xlim([t1 t2])
xlabel('tiempo')
ylabel('amplitud')

derrame2=fft(xw2);%calculo la dft de la ventana
tam=length(derrame2)/2;
dft2=derrame2(1:tam);%represento solo media señal

axes(handles.dft_axes)%represento la dft
plot(abs(dft2))
legend('DFT Ventana Triangular')
xlabel('frecuencia')
ylabel('amplitud')
```

Figura 8.14: Código de la ventana Triangular junto con su correspondiente Transformada Discreta de Fourier y sus representaciones.

En este caso, utilizo el comando bartlett para calcular la ventana triangular, posteriormente la aplico a la señal con el *gating temporal* aplicado, y la represento. Vuelvo a calcular su Transformada Discreta de Fourier y la represento.

8.13.3 Ventana temporal de Hanning

```
gating=zeros(1,L);
gating(ind1:ind2)=dft(ind1:ind2);

xw3=zeros(1,L);
w3=hanning(L);%ventana de hanning
xw3=gating.*w3';%aplico la ventana a la señal

axes(handles.ventana_axes)%represento la ventana
plot(t,abs(xw3))
legend('Ventana Hanning')
xlim([t1 t2])
xlabel('tiempo')
ylabel('amplitud')

derrame3=fft(xw3);%calculo la dft de la ventana
tam=length(derrame3)/2;
dft3=derrame3(1:tam);%represento solo media señal

axes(handles.dft_axes)%represento la dft
plot(abs(dft3))
legend('DFT Ventana Hanning')
xlabel('frecuencia')
ylabel('amplitud')
```

Figura 8.15: Cálculo de la ventana de Hanning junto con su correspondiente Transformada Discreta de Fourier y sus representaciones.

Utilizo el comando hanning para calcular dicha ventana y como en las ventanas mencionadas anteriormente calculo su Transformada Discreta de Fourier y la represento.

8.13.4 Ventana temporal de Blackman

```
gating=zeros(1,L);
gating(ind1:ind2)=dft(ind1:ind2);

xw4=zeros(1,L);
w4=blackman(L);%ventana de blackman
xw4=gating.*w4';%aplico la ventana a la señal

axes(handles.ventana_axes)%represento la ventana
plot(t,abs(xw4))
legend('Ventana Blackman')
xlim([t1 t2])
xlabel('tiempo')
ylabel('amplitud')

derrame4=fft(xw4);%calculo la dft de la ventana
tam=length(derrame4)/2;
dft4=derrame4(1:tam);%represento solo media señal

axes(handles.dft_axes)%represento la dft
plot(abs(dft4))
legend('DFT Ventana Blackman')
xlabel('frecuencia')
ylabel('amplitud')
```

Figura 8.16: Cálculo de la ventana de Blackman junto con su correspondiente Transformada Discreta de Fourier y sus representaciones.

Calculo la ventana de Blackman, utilizando el comando blackman y como en las ventanas mencionadas anteriormente calculo su Transformada Discreta de Fourier y la represento.

7.13.5 Ventana temporal Rectangular

```
gating=zeros(1,L);
gating(ind1:ind2)=dft(ind1:ind2);

xw5=zeros(1,L);
w5=rectwin(L);%ventana rectangular
xw5=gating.*w5';%aplico la ventana a la señal

axes(handles.ventana_axes)%represento la ventana
plot(t,abs(xw5))
legend('Ventana Rectangular')
xlim([t1 t2])
xlabel('tiempo')
ylabel('amplitud')

derrame5=fft(xw5);%calculo la dft de la ventana
tam=length(derrame5)/2;
dft5=derrame5(1:tam);%represento solo media señal


axes(handles.dft_axes)%represento la dft
plot(abs(dft5))
legend('DFT Ventana Rectangular')
xlabel('frecuencia')
ylabel('amplitud')
```

Figura 8.17: Cálculo de la ventana Rectangular junto con su correspondiente Transformada Discreta de Fourier y sus representaciones.

Calculo la ventana Rectangular, utilizando el comando rectwin y como en las ventanas mencionadas anteriormente calculo su Transformada Discreta de Fourier y la represento.

7.13.6 Ventana frecuencial de Kaiser-Bessel

```
ventkais=str2num(get(handles.edit_ventkais,'string'));
vent=kaiser(length(A),ventkais);
```

Figura 8.17: Cálculo de la ventana de Kaiser-Bessel.

Obtengo el valor introducido desde la interfaz mediante el comando get, a continuación calculo la ventana de Kaiser utilizando el comando káiser especificando entre paréntesis el número de muestras a enventanar y el valór introducido desde la interfaz, que es el tamaño de la ventana

9. Apéndice 2: Resumen de comandos utilizados

Comando	**Función**
handles	Selecciona un texto, un botón o una gráfica, al cual se le va a aplicar una acción, por ejemplo, para seleccionar una gráfica en la que queremos representar una figura. handles.nombre
get	Sirve para obtener el valor introducido desde la interfaz. get(handles.nombre)
set	Imprime sobre una gráfica o un texto de la interfaz gráfica un nombre, un resultado o una figura. set(handles.nombreGráfica, nombre)
csvread	Lee los datos de un archivo con formato csv. csvread(nombreDelArchivo)
csvwrite	Guarda datos en un archivo con formato csv. csvwrite(nombreDelArchivo)
fopen	Abre un archivo. fopen(nombreDelArchivo)
fclose	Cierra un archivo. fclose(nombreDelArchivo)

Comando	**Función**
fft	Calcula la Transformada Discreta de Fourier de la señal. fft(señal)
ifft	Calcula la Transformada Discreta de Fourier Inversa de la señal. ifft(señal)
haming	Calcula la ventana de Hamming, en función del número de muestras. hamming(númeroMuestras)
bartlett	Calcula la ventana Triangular, en función del número de muestras. bartlett(númeroMuestras)
hanning	Calcula la ventana de Hanning, en función del número de muestras. hanning(númeroMuestras)
blackman	Calcula la ventana de Blackman, en función del número de muestras. blackman(númeroMuestras)
rectwin	Calcula la ventana Rectangular, en función del número de muestras. rectwin(númeroMuestras)

Comando	**Función**
axes	Especifica en qué gráfica queremos que se imprima la señal a representar. axes(handles.nombreGráfica)
plot	Representa una señal en 2D. plot(nombreSeñal)
xlabel	Nombra el eje x. xlabel(nombreEje)
ylabel	Nombra el eje y. ylabel(nombreEje)
xlim	Delimita el eje x entre dos valores. xlim([valor1 valor2])
legend	Escribe la leyenda de la gráfica. legend(nombre)
abs	Calcula el valor absoluto. abs(nombre)

Comando	**Función**
uigetfile	Abre una ventana con la carpeta actual en la que podemos seleccionar un archivo y abrirlo. uigetfile(formatoDelArchivoAbrir)
fgets	Lee la primera línea de un archivo. Si lo introducimos en un bucle conseguimos leer todas las líneas del archivo. fgets(nombreArchivo)
strncmpi	Compara 'N' caracteres de dos vectores. Si son iguales devuelve un 1, si son distintos un 0. strncmpi(vector1,vector2,N)
str2num	Convierte una cadena en un número. str2num(cadena)
conj	Calcula el complejo conjugado de una señal. conj (nombreSeñal)
flipdim	Da la vuelta a todos los valores del vector. El último valor ahora es el primero y el primero el último. flipdim(vector)
getframe	Captura los frames. getframe(nombre)

Comando	**Función**
imwrite	Guarda la imagen en el formato que especifiquemos. imwrite(nombreImagen, formato)
frame2im	Convierte los frames a imagen. frame2im(nombre)
length	Calcula la longitud de un vector. length(vector)
zeros	Crea un vector de tamaño 'N' de ceros. zeros(1,N)
cla	Reinicia la gráfica seleccionada. cla(handles.nombreGráfica)